

SPRINGER
REFERENCE

John Domingue
Dieter Fensel
James A. Hendler
Editors

VOLUME 1

Handbook of Semantic Web Technologies

 Springer

Handbook of Semantic Web Technologies

John Domingue, Dieter Fensel, James A. Hendler (Eds.)

Handbook of Semantic Web Technologies

With 203 Figures and 96 Tables

 Springer

Editors

John Domingue
Knowledge Media Institute
The Open University
Walton Hall
Milton Keynes, MK7 6AA
UK

Dieter Fensel
STI Innsbruck
University of Innsbruck
Technikerstraße 21a
6020 Innsbruck
Austria

James A. Hendler
Department of Computer Science
Rensselaer Polytechnic Institute
Troy, NY 12180
USA

Library of Congress Control Number: 2011921518

ISBN 978-3-540-92912-3

This publication is available also as:

Electronic publication under ISBN 978-3-540-92913-0

Print and electronic bundle under ISBN 978-3-540-92914-7

DOI 10.1007/978-3-540-92913-0

Springer Heidelberg Dordrecht London New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

© Springer-Verlag Berlin Heidelberg 2011

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Springer is part of Springer Science+Business Media

www.springer.com

Printed on acid-free paper

Foreword

The Semantic Web community has come a long way since its beginnings in the late 1990s and early part of the twenty-first century. There are obviously many ways of categorizing the progress of the topic; however, one easy fit is through three broad phases. Pre Semantic Web, various groups across the globe who had been working in areas related to semantics began to think how semantics and ontologies could aid in certain tasks related to the emerging Web. There then followed significant research funding, which was measured in hundreds of millions of Dollars and Euros, fueling large collaboration projects first in the USA and later in Europe.

Over the last few years, we have seen a second shift. The word “if” disappeared from the vocabulary used in relation to the Semantic Web a long time ago. Now the word “when” has gone too. Primarily around the term Web of Data, we can see commercial take-up by some major players in the area of the Web, Social Networks, and Media. Right now discussions focus more on which particular branch of Semantic Web technology will grow the fastest. We have also seen in Europe a movement away from specific large research calls in the Semantic Web area to semantics being a component within other domains, including, for example, services, media, security, and even networks. This take-up by both research and practitioner communities has been largely based on the fact that the Semantic Web is built upon global standards.

This book began with a realization in the spring 2008 that we seemed to be entering a mainstream phase for our research area, and it thus would be timely to capture the main threads of work. Our first task was of course to determine the book structure.

The book is split into two volumes. The first covers foundational parts and the second applications. The core themes in the first volume comprise semantic annotation and reasoning. By definition, the Semantic Web extends the Web and thus we need semantic extensions of Web languages. This book thus includes chapters on semantic languages for the Web and also mechanisms for inserting semantics into web pages encoded in plain HTML. A main motivation for semantic annotation is that it facilitates machine reasoning. A significant research effort has taken place in relation to this topic, and we cover the primary forms as well as outlining recent efforts to support reasoning at web-scale.

Other foundational issues answer questions such as: how to automatically acquire semantic annotations? how to store these efficiently at large scale? how to query these stores over the Web? and how to express the underlying conceptual structure of the annotations? In the first volume, we also outline the architectural principles underlying the Semantic Web and conclude with some thoughts on its future.

The second volume covers the application of Semantic Web technologies to a number of real-world domains and also to other technical areas. We have seen with the Web how specific application areas can drive innovation, for example, resulting in YouTube and

Facebook for video content and social networking, respectively. The sectors covered in the volume are wide-ranging, incorporating business, science, government, media, broadcasting, and culture. As to the technical areas, we outline here how semantics can improve the management of organizational knowledge and support the use of online services as well as how search is transformed in the context of the Semantic Web.

We would like to take this opportunity to heartily thank all of the chapter authors. This book progressed in a nonlinear fashion, and we are grateful for the patience shown by all and for the rapid response during the most hectic periods. This work was supported by our advisory board and we would like to express our sincere thanks to all of them for their efforts. We are also grateful for the forbearance shown by all our colleagues at Springer.

At the time of writing, we can see that the arrival of the Semantic Web is causing inflection points in a wide variety of ways as it is adopted by new constituents in a number of niches. Real-world requirements and domain-specific opportunities are driving innovation at a growing rate. We believe that this book will serve as a useful reference point for the principles and technologies underlying the Semantic Web as it continues to enter the mainstream, and we eagerly await the results of this process.

John Domingue, Dieter Fensel, and James A. Hendler
May 2011

Advisory Board

V. Richard Benjamins

Telefonica R & D
Madrid
Spain

John Davies

Future Business Applications
and Services
BT Innovate and Design
British Telecommunications Plc
Ipswich
UK

Tim Finin

Department of Computer Science and
Electrical Engineering
University of Maryland
Baltimore, MD
USA

Fausto Giunchiglia

Department of Information and
Communication Technology
University of Trento
Povo
Italy

Mark Greaves

Vulcan Inc.
Seattle, WA
USA

Frank van Harmelen

Department of Artificial Intelligence
VU University Amsterdam
Amsterdam
The Netherlands

Ian Horrocks

Oxford University Computing Laboratory
Oxford
UK

Riichiro Mizoguchi

The Institute of Scientific and Industrial
Research
Osaka University
Ibaraki, Osaka
Japan

Mark A. Musen

The Stanford Biomedical Informatics
Research
Stanford University
Stanford, CA
USA

Guus Schreiber

Department of Computer Science
Vrije Universiteit Amsterdam
Amsterdam
The Netherlands

Daniel Schwabe

Department of Informatics
Pontifícia Universidade Católica do Rio
de Janeiro
Rio de Janeiro
Brasil

Amit P. Sheth

Department of Computer Science &
Engineering
Wright State University
Dayton, OH
USA

Chris Welty

IBM Watson Research Center
Yorktown Heights, NY
USA

Rudi Studer

FZI Research Center for Information
Technology
D-76131 Karlsruhe
Germany
and
Institut für Angewandte Informatik und
Formale Beschreibungsverfahren (AIFB)
Karlsruhe Institute of Technology (KIT)
D-76128 Karlsruhe
Germany

Table of Contents

List of Contributors xiii

Volume 1

Part 1 Foundations and Technologies

1 Introduction to the Semantic Web Technologies	3
<i>John Domingue · Dieter Fensel · James A. Hendler</i>	
2 Semantic Web Architecture	43
<i>Andreas Harth · Maciej Janik · Steffen Staab</i>	
3 Semantic Annotations and Retrieval: Manual, Semiautomatic, and Automatic Generation	77
<i>Kalina Bontcheva · Hamish Cunningham</i>	
4 Semantic Annotation and Retrieval: RDF	117
<i>Fabien L. Gandon · Reto Kruppenacher · Sung-Kook Han · Ioan Toma</i>	
5 Semantic Annotation and Retrieval: Web of Hypertext – RDFa and Microformats	157
<i>Ben Adida · Mark Birbeck · Ivan Herman</i>	
6 Semantic Annotation and Retrieval: Web of Data	191
<i>Tom Heath · Christian Bizer</i>	
7 Storing the Semantic Web: Repositories	231
<i>Atanas Kiryakov · Mariana Damova</i>	
8 Querying the Semantic Web: SPARQL	299
<i>Emanuele Della Valle · Stefano Ceri</i>	
9 KR and Reasoning on the Semantic Web: OWL	365
<i>Ian Horrocks · Peter F. Patel-Schneider</i>	

10	KR and Reasoning on the Semantic Web: RIF	399
	<i>Michael Kifer</i>	
11	KR and Reasoning on the Semantic Web: Web-scale Reasoning	441
	<i>Spyros Kotoulas · Frank van Harmelen · Jesse Weaver</i>	
12	Social Semantic Web	467
	<i>John G. Breslin · Alexandre Passant · Denny Vrandečić</i>	
13	Ontologies and the Semantic Web	507
	<i>Stephan Grimm · Andreas Abecker · Johanna Völker · Rudi Studer</i>	
14	Future Trends	581
	<i>Lyndon Nixon · Raphael Volz · Fabio Ciravegna · Rudi Studer</i>	

Volume 2

Part 2 Semantic Web Applications

15	Semantic Technology Adoption: A Business Perspective	621
	<i>V. Richard Benjamins · Mark Radoff · Mike Davis · Mark Greaves · Rose Lockwood · Jesús Contreras</i>	
16	Semantic Web Search Engines	659
	<i>Mathieu d'Aquin · Li Ding · Enrico Motta</i>	
17	eScience	701
	<i>Jun Zhao · Oscar Corcho · Paolo Missier · Khalid Belhajjame · David Newmann · David de Roure · Carole A. Goble</i>	
18	Knowledge Management in Large Organizations	737
	<i>John Davies · Paul Warren</i>	
19	eBusiness	787
	<i>Christoph Grün · Christian Huemer · Philipp Liegl · Dieter Mayrhofer · Thomas Motal · Rainer Schuster · Hannes Werthner · Marco Zapletal</i>	
20	eGovernment	849
	<i>Nigel Shadbolt · Kieron O'Hara · Manuel Salvadores · Harith Alani</i>	
21	Multimedia, Broadcasting and eCulture	911
	<i>Lyndon Nixon · Stamatia Dasiopoulou · Jean-Pierre Evain · Eero Hyvönen · Ioannis Kompatsiaris · Raphael Troncy</i>	

22 Semantic Web Services	977
<i>Carlos Pedrinaci · John Domingue · Amit P. Sheth</i>	
Glossary	1039
Index	1053



List of Contributors

Andreas Abecker

FZI Research Center
for Information Technology
Haid-und-Neu-Str. 10-14
D-76131 Karlsruhe
Germany
and
disy Informationssysteme GmbH
Erbprinzenstr. 4-12
D-76133 Karlsruhe
Germany
abecker@fzi.de
Andreas.Abecker@fzi.de

Ben Adida

Creative Commons
San Francisco
USA
and
Center for Research
on Computation and Society
Harvard University
Massachusetts Hall
Cambridge, MA 02138
USA
ben@adida.net

Harith Alani

Knowledge Media Institute
The Open University
Walton Hall
Milton Keynes, MK7 6AA
UK
h.alani@open.ac.uk

Khalid Belhajjame

School of Computer Science
University of Manchester
Oxford Road
Manchester, M13 9PL
UK
Khalid.Belhajjame@manchester.ac.uk

V. Richard Benjamins

Telefonica R&D
Ronda de la Comunicación s/n
28050 Madrid
Spain
rbenjamins@tid.es

Mark Birbeck

WebBackplane
2nd Floor, 69/85 Tabernacle Street
London, EC2A 4RR
UK
mark.birbeck@webbackplane.com

Christian Bizer

Web-based Systems Group
Freie Universität Berlin
Garystr. 21
D-14195 Berlin
Germany
chris@bizer.de

Kalina Bontcheva

Department of Computer Science
University of Sheffield
Regent Court
211 Portobello Street
Sheffield, S1 4DP
UK
K.Bontcheva@dcs.shef.ac.uk

John G. Breslin

Digital Enterprise Research Institute (DERI)
National University of Ireland
Lower Dangan
Galway
Ireland
and
Electrical and Electronic Engineering
School of Engineering and Informatics
National University of Ireland
University Road
Galway
Ireland
john.breslin@nuigalway.ie

Stefano Ceri

Dipartimento
di Elettronica e Informazione
Politecnico di Milano
Via Ponzio 34/5
20133 Milano
Italy
stefano.ceri@polimi.it

Fabio Ciravegna

Department of Computer Science
University of Sheffield
Regent Court
211 Portabello Street
Sheffield, S1 4DP
UK
f.ciravegna@dcs.shef.ac.uk

Jesús Contreras

iSOCO
Av. del Partenón
16-18, 1° 7^a Campo de las Naciones
28042 Madrid
Spain
jcontreras@isoco.com

Oscar Corcho

Ontology Engineering Group
Departamento de Inteligencia Artificial
Facultad de Informática
Universidad Politécnica de Madrid
Campus de Montegancedo s/n
Boadilla del Monte,
28660 Madrid
Spain
ocorcho@fi.upm.es

Hamish Cunningham

Department of Computer Science
University of Sheffield
Regent Court
211 Portobello Street
Sheffield, S1 4DP
UK
hamish@wrongonomy.net
hamish@dcs.shef.ac.uk

Mariana Damova

Ototext AD
Sirma Group Corp
135 Tsarigradsko Chaussee
Sofia 1784
Bulgaria
mariana.damova@ototext.com

Stamatia Dasiopoulou

Informatics and Telematics Institute (ITI)
Centre for Research and Technology
Hellas (CERTH)
6Km Charilaou-Thermi Road
P.O. Box 60361
57001 Thermi-Thessaloniki
Greece
dasiop@iti.gr

John Davies

Future Business Applications
and Services
BT Innovate and Design
British Telecommunications Plc
Ipswich
UK
john.nj.davies@bt.com

Mike Davis

Ovum
119 Farringdon Road
London, EC1R 3ER
UK
mike.davis@ovum.com

Mathieu d'Aquin

Knowledge Media Institute
The Open University
Walton Hall
Milton Keynes, MK7 6AA
UK
m.daquin@open.ac.uk

David de Roure

Oxford e-Research Centre
University of Oxford
7 Keble Road
Oxford, OX1 3QG
UK
dder@ecs.soton.ac.uk

Emanuele Della Valle

Dipartimento
di Elettronica e Informazione
Politecnico di Milano
Via Ponzio 34/5
20133 Milano
Italy
emaunele.dellavalle@polimi.it

Li Ding

Tetherless World Constellation,
Computer Science Department
Rensselaer Polytechnic Institute
110 8th St.
Troy, NY 12180
USA
dingl@cs.rpi.edu

John Domingue

Knowledge Media Institute
The Open University
Walton Hall
Milton Keynes, MK7 6AA
UK
j.b.domingue@open.ac.uk

Jean-Pierre Evain

European Broadcasting Union (EBU)
L'Ancienne-Route 17A
CH-1218 Grand-Saconnex
Switzerland
evain@ebu.ch

Dieter Fensel

STI Innsbruck
University of Innsbruck
Technikerstraße 21a
6020 Innsbruck
Austria
dieter.fensel@sti2.at

Fabien L. Gandon

INRIA-Edelweiss
2004 rt des Lucioles BP93
06902 Sophia Antipolis
France
Fabien.Gandon@sophia.inria.fr

Carole A. Goble

School of Computer Science
University of Manchester
Oxford Road
Manchester, M13 9PL
UK
carole.goble@manchester.ac.uk

Mark Greaves

Vulcan Inc.
505 Fifth Ave South, Suite 900
Seattle, WA 98104
USA
markg@vulcan.com

Stephan Grimm

FZI Research Center
for Information Technology
Haid-und-Neu-Str. 10-14
D-76131 Karlsruhe
Germany
grimm@fzi.de

Christoph Grün

Electronic Commerce Group
Institute for Software Technology
and Interactive System
Vienna University of Technology
Favoritenstrasse 9-11/188
1040 Vienna
Austria
christoph.gruen@ec.tuwien.ac.at

Sung-Kook Han

Semantic Technology Institute
Universität Innsbruck
6020 Innsbruck
Austria
sung-kook.han@sti2.at

Andreas Harth

AIFB Karlsruhe Institute of Technology
(KIT)
Englerstr. 11
D-76131 Karlsruhe
Germany
harth@kit.edu

Tom Heath

Talis Systems Ltd
Knights Court
Solihull Parkway
Birmingham Business Park
Birmingham, B37 7YB
UK
tom.heath@talis.com

James A. Hendler

Department of Computer Science
Rensselaer Polytechnic Institute
Troy, NY 12180
USA
hendler@cs.rpi.edu

Ivan Herman

World Wide Web Consortium and CWI
P.O. Box 94079
Kruislaan 413
1090 GB Amsterdam
The Netherlands
ivan@w3.org

Ian Horrocks

Oxford University Computing Laboratory
Wolfson Building
Parks Road
Oxford, OX1 3QD
UK
Ian.Horrocks@comlab.ox.ac.uk

Christian Huemer

Business Informatics Group
Vienna University of Technology
Karlsplatz 13
1040 Vienna
Austria
christian.huemer@tuwien.ac.at

Eero Hyvönen

Semantic Computing Research Group
(SeCo)
Aalto University
and
University of Helsinki
P.O. Box 15500
FI-00076 Aalto
Finland
evain@ebu.ch

Maciej Janik

Institute for Web Science and
Technologies - WeST
University of Koblenz-Landau
Universitätsstraße 1
56070 Koblenz
Germany
janik@uni-koblenz.de

Michael Kifer

Department of Computer Science
State University of New York
at Stony Brook
Stony Brook, NY 11794-4400
USA
kifer@cs.stonybrook.edu

Atanas Kiryakov

Ontotext AD
Sirma Group Corp
135 Tsarigradsko Chaussee
Sofia 1784
Bulgaria
naso@ontotext.com

Ioannis Kompatsiaris

Informatics and Telematics Institute (ITI)
Centre for Research and Technology
Hellas (CERTH)
6Km Charilaou-Thermi Road
P.O. Box 60361
57001 Thermi-Thessaloniki
Greece
ikom@iti.gr

Spyros Kotoulas

Artificial Intelligence Department
VU University Amsterdam
De Boelelaan 1081a
1081 HV Amsterdam
The Netherlands
kot@few.vu.nl

Reto Krummenacher

Semantic Technology Institute
Universität Innsbruck
6020 Innsbruck
Austria
reto.krummenacher@sti2.at

Philipp Liegl

Business Informatics Group
Vienna University of Technology
Karlsplatz 13
1040 Vienna
Austria
philipp.liegl@tuwien.ac.at

Rose Lockwood

Technology Industry Analyst
24 Harrington Square
London, NW1 2JJ
UK
semanticrose@hotmail.com

Dieter Mayrhofer

Business Informatics Group
Vienna University of Technology
Karlsplatz 13
1040 Vienna
Austria
dieter.mayrhofer@tuwien.ac.at

Paolo Missier

School of Computer Science
University of Manchester
Oxford Road
Manchester, M13 9PL
UK
missier@cs.man.ac.uk

Thomas Motal

Electronic Commerce Group
Institute for Software Technology
and Interactive System
Vienna University of Technology
Favoritenstrasse 9-11/188
1040 Vienna
Austria
thomas.motal@tuwien.ac.at

Enrico Motta

Knowledge Media Institute
The Open University
Walton Hall
Milton Keynes, MK7 6AA
UK
e.motta@open.ac.uk

David Newmann

School of Electronics
and Computer Science
University of Southampton
Southampton, SO17 1BJ
UK
drn@ecs.soton.ac.uk

Lyndon Nixon

Semantic Technology Institute (STI)
International
Amerlingstrasse 19/35
1060 Vienna
Austria
lyndon.nixon@sti2.org

Kieron O'Hara

Intelligence, Agents, Multimedia Group
School of Electronics
and Computer Science
University of Southampton, Highfield
Southampton, SO17 1BJ
UK
kmo@ecs.soton.ac.uk

Alexandre Passant

Digital Enterprise Research Institute
(DERI)
National University of Ireland
Lower Dangan
Galway
Ireland
alexandre.passant@deri.org

Peter F. Patel-Schneider

Alcatel-Lucent Bell Labs Research
Murray Hill
New Jersey
USA
pfps@research.bell-labs.com

Carlos Pedrinaci

Knowledge Media Institute
The Open University
Milton Keynes, MK7 6AA
UK
c.pedrinaci@open.ac.uk

Mark Radoff

Ovum
119 Farringdon Road
London, EC1R 3ER
UK
mark.radoff@ovum.com

Manuel Salvadores

Intelligence, Agents, Multimedia Group
School of Electronics
and Computer Science
University of Southampton
Highfield
Southampton, SO17 1BJ
UK
ms8@ecs.soton.ac.uk

Rainer Schuster

Electronic Commerce Group
Institute for Software Technology
and Interactive System
Vienna University of Technology
Favoritenstrasse 9-11/188
1040 Vienna
Austria
Rainer.Schuster@tuwien.ac.at

Nigel Shadbolt

Intelligence, Agents, Multimedia Group
School of Electronics
and Computer Science
University of Southampton, Highfield
Southampton, SO17 1BJ
UK
nrs@ecs.soton.ac.uk

Amit P. Sheth

Department of Computer Science
& Engineering
Wright State University
3640 Colonel Glenn Hwy.
Dayton, OH 45435-0001
USA
amit.sheth@wright.edu

Steffen Staab

Institute for Web Science
and Technologies - WeST
University of Koblenz-Landau
Universitätsstraße 1
56070 Koblenz
Germany
staab@uni-koblenz.de

Rudi Studer

FZI Research Center for Information
Technology
D-76131 Karlsruhe
Germany
and
Institut für Angewandte Informatik und
Formale Beschreibungsverfahren (AIFB)
Karlsruhe Institute of Technology (KIT)
D-76128 Karlsruhe
Germany
rudi.studer@kit.edu
studer@aifb.uni-karlsruhe.de

Ioan Toma

Semantic Technology Institute
Universität Innsbruck
6020 Innsbruck
Austria
ioan.toma@sti2.at

Raphaël Troncy

EURECOM
2229 Route des Crêtes
06560 Sophia Antipolis
France
raphael.troncy@eurecom.fr

Frank van Harmelen

Department of Artificial Intelligence
VU University Amsterdam
De Boelelaan 1081a
1081 HV Amsterdam
The Netherlands
Frank.van.Harmelen@cs.vu.nl

Johanna Völker

KR & KM Research Group
University of Mannheim
B6, 26
D-68159 Mannheim
Germany
voelker@informatik.uni-mannheim.de

Raphael Volz

Volz Innovation GmbH
Postfach 45
76597 Loffenau
Germany
rv@volzinnovation.com

Denny Vrandečić

Institut für Angewandte Informatik und
Formale Beschreibungsverfahren – AIFB
Universität Karlsruhe
D-76128 Karlsruhe
Germany
denny.vrandecic@kit.edu

Paul Warren

Programme Manager
Eurescom GmbH
Wieblingen Weg 19/4
D-69123 Heidelberg
Germany
warren@eurescom.eu

Jesse Weaver

Tetherless World Constellation
Rensselaer Polytechnic Institute
Troy
USA
weavej3@rpi.edu

Hannes Werthner

Electronic Commerce Group
Institute for Software Technology
and Interactive System
Vienna University of Technology
Favoritenstrasse 9-11/188
1040 Vienna
Austria
hannes.werthner@tuwien.ac.at

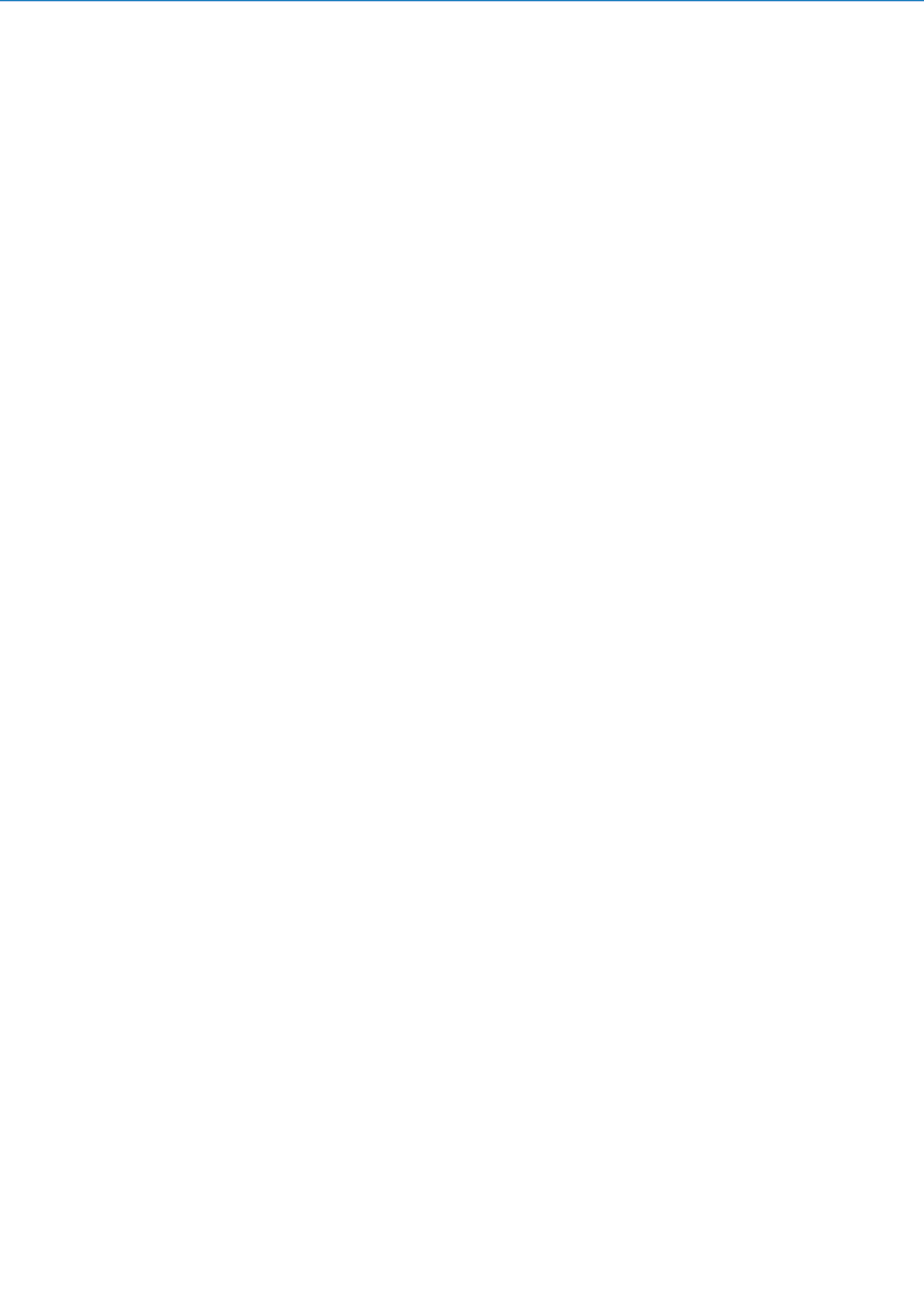
Marco Zapletal

Electronic Commerce Group
Institute for Software Technology
and Interactive System
Vienna University of Technology
Favoritenstrasse 9-11/188
1040 Vienna
Austria
marco.zapletal@tuwien.ac.at

Jun Zhao

Department of Zoology
Oxford University
South Parks Road
Oxford, OX1 3PS
UK
jun.zhao@zoo.ox.ac.uk

Foundations and Technologies



1 Introduction to the Semantic Web Technologies

John Domingue¹ · Dieter Fensel² · James A. Hendler³

¹The Open University, Milton Keynes, UK

²University of Innsbruck, Innsbruck, Austria

³Rensselaer Polytechnic Institute, Troy, NY, USA

1.1	Introduction	4
1.2	What Is the Web?	5
1.2.1	The Problem to Be Solved	5
1.2.2	Principles of the Web	6
1.2.3	Web Architecture	7
1.2.4	What Are the Problems with the Web?	8
1.3	What Are Semantics?	9
1.3.1	Semantics, the Science of (Meaning) ²	10
1.3.2	Form	12
1.3.3	Logic	12
1.3.4	Semantic Web Languages	14
1.3.5	The Tower of Babel	17
1.3.6	Substance	21
1.4	Semantics and the Web	22
1.4.1	The Semantic Web as a Layer over Text	22
1.4.2	Semantic Web as a Database	23
1.4.3	Semantic Web as a Platform for Agents	24
1.5	Brief History	25
1.5.1	Increasing Research Interest	26
1.6	Related Resources	27
1.6.1	Semantic Web Events	27
1.6.1.1	Conferences	27

1.6.1.2	Summer Schools and Tutorials	28
1.6.1.3	Semantic Web Journals and Magazines	29
1.6.1.4	Semantic Websites	29
1.6.1.5	Sources Introducing the Semantic Web	30
1.6.1.6	Books	30
1.7	<i>Selected Successes in the Commercial Sphere</i>	30
1.7.1	Oracle	31
1.7.2	Facebook's Open Graph Protocol	31
1.7.3	Google Buys Metaweb	32
1.7.4	BBC Football World Cup 2010 Website	32
1.7.5	Apple Buys SIRI	33
1.8	<i>Future</i>	34

1.1 Introduction

- ▶ The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation [6].

For newcomers to the Semantic Web, the above definition taken from the article, which is often taken as the starting point for the research area, is as good a starting point as any. The goal of the Semantic Web is in some sense a counterpoint to the Web of 2001. That Web was designed as a global document repository with very easy routes to access, publish, and link documents, and Web documents were created to be accessed and read by humans.

The Semantic Web is a machine-readable Web. As implied above, a machine-readable Web facilitates human–computer cooperation. As appropriate and required, certain classes of tasks can be delegated to machines and therefore processed automatically. Of course, the design possibilities for a machine-readable Web are very large, and a number of design decisions were taken in developing the Semantic Web as it is seen today. The trade-offs in the design space are discussed later on in this chapter and also in the rest of the book. Two of the most significant are worth mentioning up front though. Firstly, as captured in the quote above, the Semantic Web is an extension of the Web. In particular, the Semantic Web builds upon the principles and technologies of the Web. It reuses the Web’s global indexing and naming scheme, and Semantic Web documents can be accessed through standard Web browsers as well as through semantically aware applications. A global naming scheme means that in principle every semantic concept has a unique identifier, although in practice identity resolution is still a research area and the Semantic Web language OWL contains a specific relation to deal with this issue.

A second design choice is related to the fact that the Web is a shared resource, and therefore, within a machine-readable Web, meaning should be shared too. To this end, the Semantic Web incorporates the notion of an ontology, which by definition is a shared machine-readable representation (see ▶ Sect. 1.3.6). Through ontologies and ontology-related technologies, the meaning of and relationships between concepts within published Web pages can be processed and understood by software-based reasoners.

After about a decade of dedicated Semantic Web research, we are now entering a new phase for the technology. In short, it can now be claimed that the Semantic Web has arrived. There are a number of indicators to this. For example, semantic search engines now claim to index many millions of Semantic Web documents. Of course, this number of documents is small when compared to the size of the overall Web, but the trend resembles the early days of the Web, and if one counts the contained semantic statements (triples – see ▶ Sect. 1.3.4), then the number is estimated to be over a hundred billion triples.

Later in this chapter and also in most of the other chapters of this book, evidence is given to the take-up of Semantic Web technology. Semantics can be seen being deployed in a wide variety of settings including enterprise, government, media, and science arenas.

We are thus at a tipping point in the timeline of the Semantic Web where the technology can be seen to be moving out of research labs and into the mainstream in a nontrivial fashion.

To mark this juncture, this book describes the main technological components of the Semantic Web, the vertical areas in which the technology is being applied, and new trends in the medium and the long term. Each chapter covers general scientific and technical principles and also gives examples of application and pointers to relevant resources.

The rest of this chapter gives an introductory account of the notions of the Web and semantics from a technical perspective. Also, a brief history of the research area is discussed, given pointers to a number of general Semantic Web resources, and some highlights in terms of the deployment of semantic technology are outlined. The final section contains pointers to the future of the topic in general terms.

1.2 What Is the Web?

With over one trillion pages and billions of users, the Web is one of the most successful engineering artifacts ever created. At the end of 2009, there were 234 million websites of which 47 million were added in the year. The Web is now a rich media repository: the current upload to Flickr is equivalent to 30 billion new photos per year and YouTube now serves over one billion videos per day [50].

1.2.1 The Problem to Be Solved

As commonly known, the Web was invented by Sir Tim Berners-Lee while at CERN. The underlying problem he was tackling was how to manage and share technical information and knowledge at CERN where he was working at the time [5]. The overall scenario at the establishment contained several features, which can be found in many organizations over a certain size:

- The projects carried out were large and complex involving several different types of technologies.
- Work was carried by teams, which crossed CERN's specified departments and unit structures.
- The knowledge involved was not static but rather changed over time.
- There was a rotation of staff. Workers came and went periodically – the typical length of stay at CERN at the time was 2 years.

This scenario led to the following underlying general requirements:

- Workers needed to be able to easily find and access relevant documents containing technical knowledge.
- The content of the documents needed to be easily changeable and the changes propagated across the organization quickly.

- The structure of the document collection could not be predetermined and had to be adapted easily.

The problems faced within CERN were acknowledged at the time to be relatively common and also ones that would become prevalent across the globe in the near to medium term as aptly expressed:

- ▶ CERN meets now some problems which the rest of the world will have to face soon [5].

1.2.2 Principles of the Web

As succinctly coined in the phrase: “For a hammer everything is a nail” (originally from [43, p. 15]), one has to be careful when differentiating between technological biases and the true underlying principles for any generic framework. Nevertheless, a significant portion of the design of the Web is based upon Hypertext, which was originally coined as a term by Ted Nelson [48] and has roots going back to Doug Engelbart’s oNLine System [93] and Vannevar Bush’s Memex system [11]. Another stream of innovation for the Web is based upon communication protocols, notably TCP-IP, a spin-off of TCP [12], which provides the bottom layer of the communication protocol for the Web.

Twenty years on from the starting points above, the principles of the Web are firmly established. These principles, many of which can be traced back to the original CERN proposal, have contributed significantly to the Web’s success. These include:

- *Openness* Anyone or any organization can engage with the Web as a provider or consumer of information. Openness is an essential criterion for the success of the Web as a platform and incorporates:
 - *Accessibility* Web content can be accessed remotely from a wide variety of hardware and software platforms.
 - *Nonproprietary* The Web itself is not owned by any individual or organization, minimizing the effect cost has on participating.
 - *Consensual control* The Web structure is itself controlled and managed by an open body, the World Wide Web Consortium (W3C), which has a well-defined consensual process model for decision making.
 - *Usable* Usage of this infrastructure as a provider or user is kept as simple, smooth, and unrestricted as possible.
- *Interoperability* The Web is neutral to hardware and software platforms. A layer of protocols provides an integration mechanism, enabling heterogeneous proprietary and legacy solutions to interoperate through common interfaces.
- *Decentralized authorship and editorship* Content can appear, becoming modified, or be removed in a noncontrolled fashion. That is, the provisioning and modification of content is under the distributed control of the peers rather than being controlled by a central authority. Central control would hamper access and therefore scalability. A consequence of this principle is that an element of chaos or “untidiness” needs to be

tolerated. It is hard to imagine now, but in the early days of the Web one of the most common criticisms was that it would never take off because some Web pages could be found that were either incorrect or were below some quality threshold and also that some links were broken (two of the editors know of Computer Science professors who made this complaint).

- *Automated mechanisms are provided to route requests and responses* In order to scale, routing between requests and responses is handled in an automated fashion. Manual indexes or repositories are inherently non-scalable and costly, and immediately become outdated. The way that Web pages are accessed has changed over the past 10 years. At the beginning, one was required to know the IP-Address of the desired page and then later the URL (see below for a description). In this period, bookmark lists (especially lists of useful pages for a particular topic) were considered valuable intellectual property. Later, search engines such as AltaVista and Google raised access to the level of keywords.
- *Enabling n:m relationships* to maximize interaction. In contrast to email, where the content is targeted to specific receivers, the Web is based on anonymous distribution through publication. In principle, the information is disseminated to any potential reader, something that e-mail can only attempt to achieve through spam. The use of content for purposes not perceived by content producers facilitates serendipity on the Web and is one of the Web's key success enablers.

1.2.3 Web Architecture

The architecture of the Web is surprisingly simple for an engineering artifact with over a billion users. On the other hand, this is probably one of the main reasons for its success. From a functionality perspective, the Web provides the following:

- A *worldwide addressing schema*, which enables every document to have a unique globally addressable identifier. For the Web, this is provided by URLs (Uniform Resource Locators). A URL serves the purposes of both identifying a resource and also describing its network location so that it can be found. URIs (Uniform Resource Identifiers) encompass both URLs and URNs (Uniform Resource Names), where URNs denote the name of a resource.
- A *transport layer*, a protocol, HTTP (HyperText Transfer Protocol), which supports the remote access to content over a network layer (TCP-IP). HTTP functions as a request–response protocol in a client–server computing model. In HTTP, a Web browser typically acts as a client, while an application running on a computer host acts as a server.
- A *platform-independent interface*, which enables users to easily access any online resource. In case of the Web, it is HTML (HyperText Markup Language) and Web browsers that interpret and display the described content. HTML is thus a text and image formatting language, which is remotely served by Web host applications and used by Web browsers to display the Web content.

Integral to the makeup of the Web is the hyperlink which has its origins in the hypertext field. Hyperlinks allow a Web resource to point to any other Web resource by embedding the URL within an HTML construct (the “<a>” or anchor element). Links on the Web are unidirectional and are not verified, which means that links may break – the target Web resource may have been removed or the URL itself may be incorrect – leading to the “untidiness” mentioned earlier. However, not forcing links to be verified is widely accepted as being one of the design choices that enabled the Web to scale so quickly.

1.2.4 What Are the Problems with the Web?

The amount of information on the Web is staggering. The one trillion Web resources encompass practically every topic of human interest: from the life cycle of earthworms in New Zealand [110], to UK Pop Hits in the 1950s [66], to the Constitution of Mauritius [44].

Accessing documents can be efficient on the Web; if one knows the right keywords then extremely so – to the point where experienced users would rather search for the PDF of a paper online than get up out of their chairs and access a hardcopy on the shelf. The usefulness of document search can be seen from the fact that in December 2009 it was noted that 87.8 billion searches were conducted each month on Google [61]. As an extension to the Web, the Semantic Web has been created to solve two specific problems, which are as follows:

- *Accessing data* – the “standard Web” is limited in that:
 - *Documents are indexed and accessed via plain text*, that is, a string-based matching algorithm is used to retrieve documents according to a given request. This creates problems for ambiguous terms, for example, “Paris” can denote: the capital of France; towns in Canada, Kiribati, and the USA; a number of films including “Paris, Texas” by Wim Wenders; fictional characters including the legendary figure from the Trojan War; and a number of celebrities including the daughter of Michael Jackson, and Paris Hilton the socialite and heiress. Moreover, complex matching involving inference is not feasible without additional technology. For example, correctly answering the query: “where can I go on holiday next week for 10 days with two young children for less than 1000 Euros in total?” is not possible with current search engines.
 - The current paradigm is dominated by *returning single “best fit” documents for a search*. Often, the answer to a query is available on the Web but requires the combination and integration of the content of multiple source documents. The dominant search engines today leave this integration of content to the user.
 - *Underlying data are not available*. A significant number of websites are generated through databases but the underlying data are hidden behind the presented HTML. This phenomenon is sometimes termed “the dark Web” and significantly hinders the usability and reusability of the underlying information. A way to overcome this problem is to “Web scrape” the data by parsing the presented

HTML. This process though is error-prone and unstable with regard to changes in the way the page is displayed (e.g., if the layout or color scheme is altered). It should be noted that the concept of making legacy database data available was specified as a requirement in the original proposal from Sir Tim Berners-Lee.

- *Enabling delegation* – the Web can be viewed as a very large collection of static documents. When users browse the Web, their computers act simply as rendering devices displaying text and graphics and sometimes audio and video content. All inference and computation is left to the user. To a large extent, the computational abilities of the computational device are not used. Coupled with the above ones on users to carry out their own inferences, the sheer volume and growth of data available creates a strong need for at least some level of automation. For example, current estimates are that the 281 exabytes (10^6 TB) of information created or replicated worldwide in 2007 will grow tenfold by 2011 to 1 zettabyte (10^9 TB) per year. Delegating tasks such as the integration of information, data analysis, and sense-making to machines, at least partially, is the only way forward for users, communities, and businesses to continue to make the most of the information available on the Web.

Given the above requirements, the Semantic Web extends the Web with “meaning” supporting access to data at web-scale and enabling the delegation of certain classes of tasks. As the Web has documents at the center, the Semantic Web places data and the semantics of data at its core. An overview of the architecture of the Semantic Web is given in [🔗 Semantic Web Architecture](#).

1.3 What Are Semantics?

Computer science, since the early beginning, has been concerned with processing of data. Programming languages provide simple and complex datatypes to store data. Originally, the semantics of these data were hardwired in the programs in which they were interpreted and used. Around 50 years ago, data began to become separated from the application program to be stored in **databases**. This allowed one to reuse the same data in different programming contexts and prevented the same data management component being re-implemented across many applications. The fact that the meaning of the data was no longer hardwired directly into the application program led to mechanisms for representing the structure and semantics of the data being developed. One such extremely successful structure was the relational data model (cf. [23]). In addition to simple data that can be aligned easily with the constructs of programming languages, a growing number of documents in natural language started to be placed within computers in the 1960s. Unfortunately, relational database technology is not a very useful or efficient paradigm to store, manipulate, and query these types of documents. In consequence, the areas of **information retrieval** (cf. [41]), **information extraction** (cf. [46]), and **natural language processing** (cf. [34]) evolved in parallel. These areas are concerned with capturing the meaning contained in digital natural language documents to support

their automatic processing. A third area of computational semantics was founded around 1955 with the goal of enabling a computer to act intelligently as humans do, that is, generating **Artificial Intelligences** (cf. [54]). The field began by implementing general problem solving methods such as global search and theorem proving. However, after a short space of time, the numerical complexity of the tasks involved in intelligent problem solving made it apparent that a machine-understandable representation of the knowledge related to how a problem may be solved efficiently was required.

- ▶ Knowledge of the specific task domain in which the program is to do its problem solving was more important as a source of power for competent problem solving than the reasoning method employed [17].

The subareas of **knowledge representation** (cf. [8]) and **knowledge acquisition**, which was later called **knowledge engineering** (cf. [55]), were created to provide methods and techniques to represent human knowledge in a machine-understandable manner.

All these areas of Computer Science focus on capturing the meaning of data in a machine-processable manner and provide the historical context from which semantic technology was developed. The following briefly discusses the essential essence of semantic technology, as well as its form and substance.

1.3.1 Semantics, the Science of (Meaning)²

Semantic technology provides machine-understandable (or better machine-processable) descriptions of data, programs, and infrastructure, enabling computers to reflect on these artifacts. Now, what does machine-processable semantics really mean? Let us ask Wikipedia, the world leading resource of human knowledge. Let us specifically ask for **machine-processable semantics**. Unfortunately, there is no direct response. Okay let us ask for its three elements.

A *machine* is any *device* that uses *energy* to perform some activity [92]. Okay, one now needs to understand what a *device* is. Here, get a pointer to Wiktionary: “Any piece of *equipment* made for a particular *purpose*” [104]. By the way, only equipment that uses energy qualifies as a machine. Still, what is equipment and why does it require a *purpose*? Let us ask Wikipedia again. *Equipment* redirects to *tools*. Okay, let us check *tools*. “A *tool*, broadly defined, is an entity that interfaces between two or more domains;.... Basic tools are *simple machines*” [88]. Basic machines are somehow simple machines? Well, yes, but...? The aspect of *energy consumption* has still not been explored that distinguishes a machine from a generic *device*, and *purpose* that distinguishes a *device* from the more generic *equipment*.

- “In all such *energy transformation* processes, the total energy remains the same” [87]. What was meant by *consuming* energy? “*Energy* is a *quantity* that can be assigned to every *particle*” [87]. Here, proceedings become a bit philosophical. Trying to find out what a *quantity* is and why it is that it can be assigned to all *particles* will be resisted. Not to mention that the notion of an *assignment* should really be investigated and delved into

whether *particles* or *waves* are the final truth? It does not really help to distinguish between a *machine* and a *device*. That is, *machines* remain defined as being *machines* (more precisely, it is learnt only that basic machines are simple machines).

- “Purpose is a result, end, aim, or goal of an action *intentionally* undertaken” [95]. So what is an *intention*? “An agent’s *intention* in performing an *action* is his or her specific *purpose*” [91]. No, there will be no attempt to find out what an *agent* is.

Processable does not have a hit at Wikipedia. This saves both time and space.

“*Semantics* is the study of meaning, . . . This problem of understanding has been the subject of many formal inquiries. . . most notably in the field of *formal semantics*” [98]. Also from the same source: “The word ‘*semantics*’ itself denotes a range of *ideas*.” Fortunately only the *word*. And no, we will not try to understand what an *idea* is, since already in the narrowest sense “an *idea* is just whatever is before the *mind* when one thinks” [90]. Let us try to find out the meaning of formal semantics: “*Formal semantics* is the study of the semantics” [89]. Okay, formal semantics is the study of semantics and semantics is the study of meaning. Obviously meaning is the study of? **No**, meaning “is the end, purpose, or significance of something” [64]. So, formal semantics is the study of the study of purpose. Purpose is to remember the attribute used to distinguish a device from generic equipment (which is a machine if it consumes energy).

Naively entered here is an infinite regression of circular definitions written in natural language. This would be an opportune moment to refer to the importance of cooperation as a grounding mechanism for communication and to conduct a detailed analysis of the role of vocal and nonvocal communication mechanisms (cf. [45, 56]) in order to escape this infinite regress. However, this is not the focus here. Obviously, life is a circle and one needs to be pragmatic. Let us try to understand the essence of semantic technology through its usage starting with a number of predecessor technologies.

What is the main value of a traditional relational *database*? According to Wikipedia, “a *database* is a collection of data” and “the term *data* means groups of information” . . . “*Information* as a concept has many meanings . . .” The authors do not tell us whether information that is not viewed as a concept would have less meaning. According to Wikipedia, *meaning* also has many meanings. Still, Oracle is able to successfully sell *bases of collections of groups of information that have many meanings when viewed as a concept not mentioning the fact that already meaning has many meanings*. Moreover, Oracle makes billions of dollars per annum with this kind of rather vague business.

In a relational database, everything is represented in a table, and a row has a key and a column has a name. With this, even with a very simple machine, one can find the phone number of Mr. X if X is the value of the name column and phone number is the heading of another column. Unfortunately, with an average Web page, this is far more difficult. As mentioned earlier, hidden in various HTML tags there is a name (a random alphanumeric string similar to many others) and somewhere else a phone number (a set of integers including some special characters). A browser is required to render the information and a human reader to understand the information based on the layout of the website. This is the solution as implemented in the Web which was introduced 20 years ago. As outlined

earlier, the sheer simplicity has made the Web an incredible success story with now more than one billion users. Its simplicity also leaves room for improvement.

Semantic technology adds tags to semistructured information as database technology adds column headings to tabular information. Let us use a small example:

```
<person>
  <name>Sir Tim</name>
  <phone number>01-444444</phone number>
</person>
```

These annotations allow a computer “to understand” that *Sir Tim* is a name of a person and *01-444444* is his telephone number. In a similar fashion, programs and other computational resources can be described through semantic annotations. *This is the essence of Semantic Web technology.*

What can be seen from this example is that one needs two things to define the semantics of information: a language such as $\langle X \rangle Y \langle /X \rangle$ to define the meaning of Y , and terms such as x to denote this meaning. This is investigated in more detail in the following.

1.3.2 Form

Logic is a 1,000-year-old technology to formally capture meaning. Over this long history, especially relatively recently, a large number of logics have been developed, each suitable for a specific purpose. The focus is on a small number of these languages, in particular, on those that provide insights into the overall design issues associated with logical languages and those that have been applied in a Semantic Web context. A number of languages will be then examined that are used to express the meaning of data on the Semantic Web. Finally, there would be a discussion on open issues and problems when applying logic to the Web.

1.3.3 Logic

From an algorithmic perspective, implementing logical-reasoning systems demonstrates clearly how complex decidability and complexity are to manage (cf. [29, 35]). First, briefly described are logical paradigms in increasing levels of complexity, and then, how computer scientists identified reasonable subsets which can be handled to a certain extent.

Propositional Logic is a rather simple logic language providing propositions such as A, B, C, \dots and logic connectives such as AND and OR. All interpretations are simply the enumerations of all possible false and true assignments to these propositions. Therefore, propositional logic is decidable, although, already NP-hard.

First-Order Predicate Logic provides a richer means to define such propositions by providing terms such as $c, f(c, X), \dots$ and predicate symbols that can be applied to these terms $P(c), Q(c, f(c, X)), \dots$. Terms can make use of variables that can be existentially or all quantified (i.e., either there must exist a term fulfilling a formula or all terms must fulfill a formula). First-order predicate logic is still semi-decidable. That means,

there are complete and correct evaluation methods; however, it is not possible to guarantee that they terminate. An important feature of first-order logic is the distinction between terms and predicates, that is, one is not allowed to apply predicates or terms to predicates.

Second-Order Predicate Logic [96] and comparable languages drop this limitation (cf. [13]). Here, one can apply predicates to other predicates or entire formulae and interpret variables as sets rather than as individuals of a domain of interpretation. Unfortunately, for these languages, already unification, that is, the question of whether two terms can be substituted, is semi-decidable, which means that there is not even an approach for implementing inference in these languages. The question of how far one can make progress in simulating second-order features syntactically (statements over statements or classes that can be instances of other classes) in a semantic first-order framework has been explored in F-Logic (cf. [37]) and more generally in HiLog (cf. [13]).

In layman terms, propositional logic is reasoning about individuals. It is decidable but the effort grows exponentially with the number of individuals. First-order logic is reasoning over sets of individuals (each predicate is interpreted as a set), which is complete but does not guarantee a terminating decision procedure. Second-order logic is concerned with reasoning about sets that have elements which may again be sets. The focus of **computational logic** is on identifying subsets of logic that can be handled by computers. Unfortunately, what one gets here is not necessarily what one would need.

Most approaches in automatic theorem proving and software verification use variants of **first-order logic** to reason (cf. [53]). Here, based on the transformation of the general clause form, resolution and unification (cf. [8]) provide a complete although only semi-decidable decision procedure. Obviously, for this level of expressiveness, only incomplete reasoning requiring heuristic guidance can be achieved in the general case.

A restriction of the pragmatic complexity can be achieved by restricting first-order logic to **Horn logic** and applying Selective Linear Definite resolution [99]. There are also variants that forbid or cleverly restrict the usage of function symbols creating a decidable language – propositional logic with some additional syntactical sugar. Most work on Horn logic alters the model theory of logic by not considering all models but models that are defined through certain *minimality* criteria (this model is unique in the case where negation in the bodies of the Horn clauses is either restricted or does not exist, cf. [39]). In layman terms, this model assumes that only facts which can be inferred are true and that all other facts are false. This is called the *closed-world assumption* and originates from the database area. A well-known implementation of this paradigm is Prolog (cf. [14]). Interestingly enough, this paradigm extends the expressiveness of these syntactically restricted first-order languages beyond first-order logic as it becomes possible to express the transitive closure of a relationship.

Description Logics (cf. [3]) provide a whole family of sub-languages of first-order logic of differing complexity. Common among these languages is to restrict the formalism to *unary and binary predicates* (concepts and properties) and to restrict the usage of function symbols and logical connectors to build complex formulae. The different levels of complexity and the decidability of these languages follow from the precise definition of these restrictions. Therefore, many different languages have been defined and implemented,

many of which contain intractable worst-case behavior but which however still work for many practical applications (cf. [30]).

1.3.4 Semantic Web Languages

HTML provides a number of ways to express the semantics of data. An obvious one is the META tag [108]:

```
<META name = "Author" lang= "fr" content = "Arnaud Le Hors">
```

In the time before the wider usage of RDF, systems such as Ontobroker (cf. [19]) used the attribute of the anchor tag to encode semantic information (see the [Sect. 1.5](#)). It is also possible to interpret the semantics of HTML documents indirectly. For example, information captured in a heading tag of level one (<H1>) may be used to encode concepts that are significantly important for describing the content of a document. Still, HTML was not designed to provide descriptions of documents beyond that of informing the browser on how to render the contents. Within efforts to stretch the use of HTML to include meaning, the term semantic HTML was created – see [97] for more details on this.

The **Extensible Markup Language (XML)** [109] has been developed as a generic way to structure documents on the Web. It generalizes HTML by allowing user-defined tags. This flexibility of XML, however, reduces the possibilities for the type of semantic interpretation that was possible with the predefined tags of HTML.

The **Resource Description Framework (RDF)** (cf. [42]) is a simple data model for semantically describing resources on the Web. Binary properties interlink terms forming a directed graph. These terms as well as the properties are described using URIs. Since a property can be a URI, it can again be used as a term interlinked to another property. That is, unlike most logical languages or databases, it is not possible to distinguish the language or schema from statements in the language or schema. For example, in the statement <rdf:type, rdf:type, rdf:Property> it is stated that type is of type property. Also, unlike conventional hypertext, in RDF, URIs can refer to any identifiable thing (e.g., a person, vehicle, business, or event). This very flexible data model is obviously suitable in the context of a free and open Web; however, it generates quite a headache for logicians who wish to layer a language on top. More details on RDF can be found in [107] and in [Sect. 1.3.4 Semantic Annotation and Retrieval: RDF](#).

RDF schema (RDFS) (cf. [9]) uses basic RDF statements and defines a simple ontology language. Specifically, it defines entities such as `rdfs:class`, `rdfs:subclass`, `rdfs:subproperty`, `rdfs:domain`, and `rdfs:range`, enabling one to model classes, properties with domain and range restrictions, and hierarchies of classes and properties. RDFS is a specific RDF vocabulary for this purpose and is simply RDF plus some more definitions (statements) in RDF.

The **Web Ontology Language OWL** (cf. [16]) extends this vocabulary to a full-fledged spectrum of Descriptions Logics defined in RDF, namely, OWL Lite, OWL DL, and OWL Full. Mechanisms are provided to define properties to be inverse, transitive, symmetric, or functional. Properties can be used to define the membership of instances for classes or

hierarchies of classes and of properties. Frankly, OWL Lite is already quite an expressive Description Logic which makes the development of efficient implementations for large data sets quite challenging and, in practice, as difficult as implementing OWL DL. However, neither of these languages can make use of full RDF, that is, some valid RDF statements are not valid in Lite or DL. This is due to the fact that logic languages such as Descriptions Logics exclude meta statements, that is, statements over statements. For RDF and RDFS, this was not a problem since neither language provided mechanisms to define complex logical definitions. Spoken in a nutshell, Lite and DL define a vocabulary in RDF **and** restrict the usage of RDF. OWL Full drops these restrictions. OWL Full provides the vocabulary of OWL DL, that is, an expressive Description Logic, and allows for any valid RDF statement. For example, in OWL Full, a class can be treated simultaneously as a set of individuals and as an individual. Therefore, OWL Full is beyond the expressive scope of Description Logic and minimally requires a theorem prover type of inference such as first-order logic (i.e., is semi-decidable).

Still, OWL Full can be used as a basis to find useful restrictions (OWL DL is an example of such a restriction) and generate useful languages such as the **Simple Knowledge Organization System (SKOS)** (cf. [33]). SKOS is a data model for knowledge organization systems that uses keywords to describe resources. SKOS is defined as an OWL Full ontology, that is, it uses a sub-vocabulary of OWL Full to define a vocabulary for simple resource descriptions based on controlled structured vocabularies.

OWL2 (cf. [47]) started in 2007 to address some of the issues around OWL. In particular, OWL Lite had been defined as an overexpressive Description Logic. This hampered the implementation of Lite reasoning based on existing semantic repository technologies and also made the layering of rules on top of the language unfeasible. Specifically, there was too big a gap between RDFS and OWL Lite. In consequence, three new sub-languages were defined. OWL2EL provides polynomial time algorithms for all the standard reasoning tasks of description logic, OWL2QL enables efficient query answering over large instance populations, and **OWL2RL** restricts the expressiveness with respect to extensibility toward rule languages. OWL2RL seamlessly links with rule-based presentations of RDFS and extensions to simple rule languages (cf. [32], [52]). This is currently the route that most industrial semantic repository developers follow and will probably define together with OWL2QL the most important Semantic Web representation languages from a technological point of view.

The **Rule Interchange Format (RIF)** (cf. [36]) complements OWL with a language framework centered on the rule paradigm. Like OWL, it does not come as a single language but as a number of sub-languages. The framework incorporates RIF-BLD, which defines a simple logic-oriented rule language; RIF-PRD, which captures most of the aspects of production rule systems; and RIF-Core, which is the intersection of both these languages. This split is due to the fact that the W3C working group had to cover two very different paradigms which are only similar at the surface level: rules based on a declarative interpretation of logic (cf. [39]) and rules that model event–action systems based on the production rule paradigm (cf. [24]). The former usually have a declarative semantics in terms of a variation of a minimal Herbrand model and were an alternative

model for databases called deductive databases. The latter normally only have an operational semantics and are used to express the dynamic aspects of processes. Production rules are in essence a kind of programming language based on a blackboard architecture and event triggers. Since these production systems are no longer called expert system shells but business rule engines (suitable to implement business processes), they have gained significant commercial interest. Creating a merger of these two different paradigms was a nontrivial task. Finally, these three dialects are complemented by The Framework for Logic dialects (RIF-FLD) as a way to define new RIF dialects. RIF uses XML as the exchange syntax and unfortunately does not directly layer on top of RDF.

Since RDF is a data model, it also requires a query language. As SQL [100] is a means to express queries over relation databases, **SPARQL** (cf. [51]) is a query language for the graph-based data model of RDF. SPARQL has been developed without considering RDFS, OWL, and RIF (see [▶ Fig. 1.2](#)). More details on the query language can be found in [▶ Querying the Semantic Web: SPARQL](#).

Up to now formats to create metadata statements have been discussed, but not how to link these to existing Web content. Returning to the earlier example:

```
<person>
  <name>Sir Tim</name>
  <phone number>01-444444</phone number>
</person>
```

A way to define a concept `person` and properties such as `name` and `phone number` has been developed, but there is yet no mechanism to express that `Sir Tim` is the name of a person. Grounding or connecting metadata with documents on the Web is supported by a set of languages. **Microformats** [75] are predefined formats to add meta information to elements in HTML and XML. A well-known microformat is `hCard`, which can be used for representing people, companies, organizations, and places, using `vCard`, a file format standard for electronic business cards. These formats not only provide a language structure to present information but additionally provide domain-specific terminologies (controlled vocabularies) for this purpose. Therefore, they directly interweave structure and content. **RDFa** (cf. [1]) provides a set of XHTML attributes to include RDF metadata directly into HTML and XML documents. In contrast to Microformats, RDFa does not predefine domain-specific terminologies. **GRDDL** has been developed as a mechanism for **G**leaning **R**esource **D**escriptions from **D**ialects of **L**anguages to derive RDFa definitions from Microformats (cf. [28]) helping to integrate information from heterogeneous sources. More information on Microformats, RDFa, and GRDDL can be found in [▶ Semantic Annotation and Retrieval: Web of Hypertext – RDFa and Microformats](#). Documents are, however, only one type of data source available on the Web. In addition to being a global repository for human-readable documents, the Web is becoming more and more a platform for applications and application integration. Within the Web of Data (cf. [7] and also [▶ Semantic Annotation and Retrieval: Web of Data](#)), billions of semantically described data items have been made available for applications to consume and process. The majority of data is generated from relational databases and there has been recent

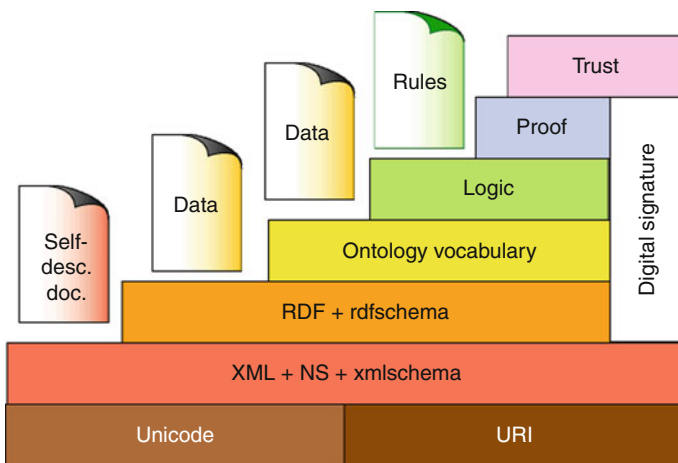
associated W3C effort, **R2RML**, to define mappings from relational models to RDF, that is, connecting databases with semantic metadata. As stated in [106], “The **mission** of the RDB2RDF Working Group is to standardize a language for mapping relational data and relational database schemas into RDF and OWL, tentatively called the RDB2RDF Mapping Language, R2RML.” Finally, one can consider the Web from a perspective of services that provide functionality either for other services or human users. Attaching semantics to services can be achieved through **Semantic Annotations for WSDL and XML Schema (SAWSDL)** (cf. [38] and also [Semantic Web Services](#)).

1.3.5 The Tower of Babel

- ▶ The Open Systems Interconnection model (OSI model) is a product of the Open Systems Interconnection effort at the International Organization for Standardization. It is a way of sub-dividing a communications system into smaller parts called layers. A layer is a collection of conceptually similar functions that provide services to the layer above it and receives services from the layer below [94].

This model is widely used in designing network architectures on a global scale. A model starts with the physical layer and ends with the application layer that provides mechanisms such as the HTTP protocol. For example, in the Internet stack, the Internet protocol components IP and TCP are at levels 3 and 4. Sir Tim Berners-Lee started a similar conceptual effort to structure the Semantic Web (see [Fig. 1.1](#)).

At the lowest level, Unicode is seen as a means to encode text, URIs to refer to resources, and XML with its namespace and schema mechanisms to provide syntactic descriptions of structured objects. On top of this, he envisioned five layers of semantics: RDF, OWL, RIF, and layers for proof and trust. This type of layering has two major functions: preventing an



■ Fig. 1.1

The Semantic Web Layer Cake – 1

upper layer from re-implementing functionality provided by a layer below and allowing an application that only understands a lower layer to at least interpret portions of definitions at a higher layer.

- ▶ The design should be such that agents fully aware of a layer should be able to take at least partial advantage of information at higher levels. For example, an agent aware only of the RDF and RDF Schema semantics might interpret knowledge written in OWL partly, by disregarding those elements that go beyond RDF and RDF Schema. Of course, there is no requirement for all tools to provide this functionality; the point is that this option should be enabled [2].

For example, OWL should not define a new `owl:Class` statement but rather reuse the already provided `rdfs:Class` statement.

Ideally, an RDFS-aware agent may not understand a property restriction for an OWL class but at least it would understand some of the elements of a class definition in OWL. Unfortunately, this is not the case.

- ▶ The rationale for having a separate OWL class construct lies in the restrictions on OWL DL (and thus also on OWL Lite), which imply that not all RDFS classes are legal OWL DL classes [105].

That is, OWL does **not** layer properly on top of RDF and RDFS (cf. [49]). This also breaks the second compatibility of [2]:

- ▶ Downward compatibility. Agents fully aware of a layer should also be able to interpret and use information written at lower levels. For example, agents aware of the semantics of OWL can take full advantage of information written in RDF and RDF Schema.

Unfortunately, this is also **not** the case! Even worse, these faults in layering OWL on top of RDF properly are not due to the fact that our colleagues involved in the language were incompetent. It actually reflects a fundamental problem associated with layering logic on top of the RDF. As outlined before:

- RDF allows arbitrary statements over statements and reflects an intrinsic property of the Web.
- OWL Lite and OWL DL as first-order logic pedantically distinguish statements in the language from statements about the language which are kept strictly separated.

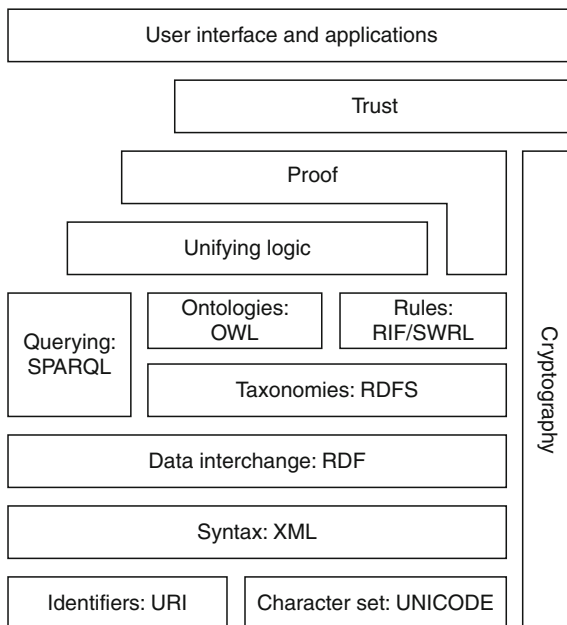
Obviously, this creates conflict and only experience can show how this fundamental problem can be resolved in a pragmatic manner that best fits practical needs. Note that statements over statements (and, e.g., statements over logic connectives such as AND and OR inside the language) is even beyond second-order logic and requires self-referenciability with all its paradox conclusions, such as allowing to express an RDF statement that states that it is not an RDF statement. A radical outcome could be that logic is not well suited for the Semantic Web, however; what else could play this role?

Another issue of layering is internal to OWL. As previously mentioned, OWL Lite was too powerful a Description Logic to be really distinguishable from OWL DL in computational terms. Here, a central design concern of OIL Light (cf. [20]) was ignored. The possibility to establish a coherent extension to RDFS that also enabled the possibility to

layer rules on top was missed. Meanwhile, with the less expressive sub-language profiles in OWL2, this has now been repaired, and obviously OWL Lite will be less than a footnote in the development of the Semantic Web. [► KR and Reasoning on the Semantic Web: OWL](#) contains a comprehensive overview of OWL.

An early layering proposal for a rule language on the Web was SWRL (cf. [31]). SWRL neatly layered a rule language on top of OWL, that is, as an extension of the already available OWL vocabulary. Unfortunately, this layering did not capture the essence of either Description Logics or of rule languages. Both are defined as fragments of first-order logic to reduce the computational complexity of executing inference. When simply combining them, this feature gets lost. As a result, one has a syntactic restriction of first-order logic without any gain in computational terms. Only when one restricts the rules to DL-safe rules is decidability restored. Simply, OWL Lite was too powerful a Description Logic to be used as a starting point for a feasible rule language. This problem is actually reflected in an update of the layer cake, as presented in [► Fig. 1.2](#). You may notice in this figure that *proof* is no longer a proper layer, that a query language is developed as an alternative to the logic stack, and finally that there is a wish for the Holy Grail, a unifying logic.

In conclusion, RIF was developed in parallel to OWL. Actually, it views XML as an exchange syntax and, as mentioned previously, is not defined as a layer on top of RDF (see [► KR and Reasoning on the Semantic Web: RIF](#)). It is therefore somewhat isolated from the other languages associated with the Semantic Web.



■ Fig. 1.2

The Semantic Web Layer Cake – 2

As already mentioned earlier, most rule languages slightly alter the semantics of first-order logic by not using all possible models but a specific (minimal) model. This comes along with what is called the **closed-world assumption**. If a fact is not evaluated to be true in this model, it is assumed to be false. This goes beyond the expressive power of first-order logic (which OWL is based on). Here, simply a truth value will not be assigned to it, since it is not restricted to a specific model. That is, it is not inferred that a fact is false from the situation where a fact is not known to be true in a specific model. This is termed the **open-world assumption**. As the Web is an open world, an open-world assumption sounds like a suitable proposition. However, with the same rationale, one could also argue for reasoning based on the closed-world assumption in relation to the portion of the Web one is investigating. This difference between rule and Description Logic languages is also reflected in the way they interpret integrity constraints, such as the domain and range restrictions of properties. When the value of a property is found and it is not known that it is a member of its range, it is assumed that there must be a mistake. The violation of a constraint is indicated over the range of the property. This is how most rule languages work. It is **not** known that a fact holds and one therefore assumes its **negation**. OWL does the opposite. OWL would infer that this value must be an element of the set defining the range of the property since the integrity constraint is requesting this. Frankly, it is hard to tell which type of reasoning is most suitable for the Web. Therefore, the designer of RDFS took a wise decision:

- ▶ For example, an RDF vocabulary might describe limitations on the types of values that are appropriate for some property, or on the classes to which it makes sense to ascribe such properties. The RDF Vocabulary Description language provides a mechanism for describing this information, but does not say whether or how an application should use it. For example, while an RDF vocabulary can assert that an `author` property is used to indicate resources that are instances of the class `Person`, it does not say whether or how an application should act in processing that range information. Different applications will use this information in different ways. For example, data checking tools might use this to help discover errors in some data set, an interactive editor might suggest appropriate values, and a reasoning application might use it to infer additional information from instance data. RDF vocabularies can describe relationships [9].

Already from this statement, you can trace the branching of OWL and RIF.

RIF has the fundamental problem of covering rule languages based on very different paradigms incorporating either a declarative or an operational flavor. It is of no surprise that RIF is not a single language but, within its first version, provides three languages. OWL now provides at least six different dialects. Thus in total, one has more than ten Semantic Web languages, and RIF additionally contains a framework for defining more. This language fragmentation is quite dangerous as it may significantly hamper information interoperability between Semantic Web applications and also significantly increase the effort to implement them.

As a final example, let us examine the layering of SKOS. First of all, SKOS uses RDF and OWL. Therefore, it should be assumed that SKOS is layered on top of OWL. However, it is

simpler than OWL. OWL is supposed to provide a language for defining ontologies, and SKOS is a way to define simple “taxonomies.” Therefore, it does not extend OWL but rather defines a small extension of a heavily constrained restriction of OWL. In general, one would naively expect to define OWL as an extension of SKOS. Not to mention that SKOS is agnostic in regard to whether its restricted version of OWL is interpreted as OWL DL or OWL Full. In the end, the Semantic Web is closer to the tower of Babel than to a coherently layered network protocol stack, and Yahweh, the enemy of global communication, may succeed again [103]. Moreover, currently there is no theoretical technique that one can apply to select one of these languages as the “right” language. Maybe the wisdom of the crowd or swarm intelligence may solve this issue in terms of impact. One may also worry a little less given the fact that holy logic also has a similar problem in that rule languages syntactically restrict and semantically extend first-order logic. What a layering!

1.3.6 Substance

For defining machine-processable metadata, a formal language for definitional purposes is required and also for linking to content available on the Web. In addition, terms are needed to actually write down metadata statements.

The simplest technique is to support keyword lists taken from a natural language. This is often called a **tag**. These tags can be freely chosen or predefined by a **controlled vocabulary** (this type of tagging is also called “subject indexing”) [86]. Folksonomies as used at Web 2.0 websites are an example of the former. Users can freely define tags, and tag clouds indicate the most popular term for a subject [101]. In library science, controlled vocabularies are widely used. However, it is not enough to simply control the vocabulary; one must also control its usage. There are various studies indicating that people will choose different terms to annotate a resource and that these terms may not be necessarily useful when a user is searching for the resource and is not familiar with the vocabulary. A controlled vocabulary can be based on a **thesaurus** such as WordNet [85] that groups nouns, verbs, adjectives, and adverbs into sets of synonyms (i.e., concepts).

The next step is to use a **taxonomy** [102], which is a classification schema arranged in a hierarchical structure. Simple taxonomies can be formalized in RDFS that provides hierarchies of classes and properties. When adding formal definitions to state that a certain value of a property must be fulfilled in order to classify it as an instance of a certain class, one can use language elements of OWL or RIF. **Ontologies** (cf. [22]) are discussed in detail in [▶ *Ontologies and the Semantic Web*](#), and so would be discussed lightly here. A very common definition of ontologies attributed to Gruber [25] is that *an ontology is a formal, explicit specification of a shared conceptualization*. Each of these attributes denotes the following:

- **Formal** – the specification is represented in a formal language which is machine processable. For the Semantic Web, this means one of the standard representation languages such as RDF or OWL.

- **Explicit** – as appropriate underlying assumptions are written down. There is a design trade-off as to how much of a domain should be contained in a specification: the level of granularity (how fine-grained) and the level of abstraction or genericity. The dimensions of this design space include:
 - *Usability* – this includes both being understandable by developers or a targeted community and also the match between the conceptualization and the requirements associated with the tasks and software applications that the ontology is used within.
 - *Reusability* – minimizing dependence with any specific task, software component, or other ontology.
- **Specification** – an ontology is a description of the artifact and is independent from the entity described. This is most meaningful when the target domain covers IT resources such as software components.
- **Shared** – an ontology only makes sense if it is shared by a community of use. The purpose and benefit of ontologies in a Semantic Web context is that they support interoperability between the designer or producer of a resource and the (software-underpinned) user. A set of formal statements hidden on a single machine does not fulfill the definition nor the purpose of an ontology.
- **Conceptualization** – an abstract simplified view of a domain of interest which is required for some task or purpose. Following from this, one thus expects ontologies to have a level of coherence and completeness with respect to a certain domain.

Note that one views all the metadata formats discussed earlier as ontologies which vary in the level of formalization. Examples of widely used ontologies are Dublin Core [65] for describing resources through properties such as title, creator, subject, publisher, etc., and **Friend Of A Friend (FOAF)** [68] that defines a set of properties such as name, e-mail address, home page, and interests to describe and link people.

1.4 Semantics and the Web

Over the last 10 years, there have been a number of ways in which different communities have envisioned how semantics and the Web can be combined. Each of these has in part been due to the research areas from which the communities originally came from and partly related to a particular conceptualization of what a semantically enhanced Web would look like. It is worth reflecting on these in order to appreciate the Semantic Web as a research topic.

1.4.1 The Semantic Web as a Layer over Text

A number of the issues raised here are covered in [🔗 *Semantic Annotations and Retrieval: Manual, Semiautomatic, and Automatic Generation*](#) technically in depth. However, it is still worth exploring some of the underlying issues. Even 10 years ago, when the Semantic Web began to take off, the Web was large (7 million unique sites [78]). Moreover, more so

than now it could be characterized as a large collection of text. From the beginning of the Semantic Web as a research project, there was a view that the key problem was how to connect with the current Web as a text resource, that is, how to transform a Web of millions or billions of text documents into a well-structured and well-defined repository of semantically described assets.

Relatively quickly a number of issues emerged. Text on the Web is not the same as text found in non-Web documents (e.g., company reports) which previous Natural Language Processing (NLP) research had focused on. Specifically, on the Web:

- Text can be shorter, comprising short phrases or single words.
- Text can be ungrammatical.
- The interpretation of text can rely on the underlying HTML-based structure, for example, laying out multiple columns in a table or the font used.

Because of the above, most successful NLP approaches to the Semantic Web rely on no or only shallow parsing.

As well as the input to the systems being different, differences in the required output also led to a stream of research. Information Extraction (IE) technologies, able to identify known entities, such as people, places, and organizations, had initial successes when applied to the Web, but these systems tended to produce unconnected entities, for example, that “John Lennon” is a person and “Imagine” is a song, missing out the relation between the two.

A more general issue associated with the above is the generic way in which NLP and ontology-based-reasoning components were integrated in applications. For the most part, these components were placed as black boxes, which were pipelined together. Only recently, in projects such as LarKC [74], has significant effort been put into combining algorithms associated with the two research areas.

A final issue related to the Semantic Web and NLP has been how to relate the (newly produced) semantic data to the original text. Trade-offs in this design space included:

- Minimizing the additional data added to the original Web page
- Facilitating the reuse of the data accumulated
- Supporting maintenance when the original Web page is altered

As mentioned above, some of the issues described here are outlined in [Semantic Annotations and Retrieval: Manual, Semiautomatic, and Automatic Generation](#).

1.4.2 Semantic Web as a Database

The Semantic Web as a research area saw the coming together of a number of communities including Artificial Intelligence (from agents, knowledge modeling, and logic) and the Web. For the most part, though, the research overlap between the Semantic Web and databases was minimal. This could be seen as somewhat surprising as the Web of Data is now a widely used term, but, in the early days, the emphasis was on creating knowledge structures as a platform for agents (see below).

The emergence of linked data as described in detail in [▶ *Semantic Annotation and Retrieval: Web of Data*](#) and the use of linked data in initiatives such as those described in [▶ *eGovernment*](#) have given rise to a stream of research which brings together the Semantic Web and database communities. RDF stores are now seen from the academic and industrial sectors, which can be deployed in settings where performance is a key issue. For example, below is outlined how an RDF triple store was used to support the BBC Sport's pages during the 2010 World Cup, which received millions of page requests per day [57].

Commercial successes such as mentioned above have now led to a more detailed discussion with the overall goal of bringing the logic and data close together. The main research issues that are currently beginning to emerge include the following:

- Which particular database techniques (e.g., partitioned hashes, column tables) are most applicable to high-performance RDF storage?
- How to structure benchmarks for large-scale repositories? Including what are the correct dimensions?
- When and where should reasoning be handled? For example, materialization (the precomputation and storage of inferred triples) is an expensive process which may not contribute to desired results.

These issues are discussed in detail in [▶ *Storing the Semantic Web: Repositories*](#). Another contribution to this debate is the Billion Triple Challenge run in conjunction with the International Semantic Web Conference (see below in Related Resources) [60]. Finally, Orri Erling has an interesting database-centric blog on this in [76].

1.4.3 Semantic Web as a Platform for Agents

From the beginning, the Semantic Web was seen as a necessary platform for supporting agents which could carry out tasks on behalf of human users. Within the seminal Semantic Web paper [6], a scenario is presented at the start where a Semantic Web agent books an important medical appointment checking the online diaries of a woman, her two grown-up children, and a number of hospitals satisfying geographic and quality constraints. The motivation for creating the Semantic Web is based on the functionality provided by software agents, which rely on the combination and exchange of content from diverse sources. The Semantic Web would allow agents to read the content of pages because the data are coded in a machine-readable representation. The underlying ontological basis for the data supports semantic interoperability by coding meaning in a way that supports semantic mediation.

Given the early motivations, however, the amount of agent research based on Semantic Web technology has been relatively small. There were two main reasons for this. Firstly, more emphasis than initially envisioned was required for creating a robust, usable, and scalable data layer. Also the majority of agent research was founded on FIPA protocols [67] rather than the stack of Web standards. Reevaluating the Semantic Web agent vision in light of newer phenomena such as the Web of Data and the Social Web (see [▶ *Social Semantic Web*](#)) would be an interesting research exercise.

Research in Semantic Web Services, covered in [▶ Semantic Web Services](#), has also been seen as a means to provide an infrastructure for Semantic Web agents, but this has not been widely pursued.

1.5 Brief History

It is hard to know who first had the idea of creating a language on the World Wide Web that could be used to express the domain knowledge needed to improve Web applications. By the mid-1990s, before most people even knew the Web existed, several research groups were playing with the idea that if Web markup (which was all primarily HTML) contained some machine-readable “hints” to the computer, then one could do a better job of Web tasks like search, query, and faceted browsing. It is important to note that at that time, the potential power of the Web was still being debated, and there were many who were sure it would fail (see the [▶ Sect. 1.2.2](#)).

However, by 1997 or so, it was clear that the Web was going to be around for some time, and there was a burst of energy going on. Various researchers were publishing algorithms, suggesting that different approaches could be used for searching the Web rather than the traditional AI approaches, and it was around this time that Sergey Brin and Larry Page published their famous “PageRank” paper [10], which led to the creation of Google and the growth of the modern search engine. This historical event is mentioned here as it is sometimes said that the Semantic Web was created to improve search. This is partly true, but it is important to note that search as known back then, pre-Google, was not the same as the current keyword search that powers so much of the modern Web today.

At this time, the first “real” refereed publications were also seen coming about machine-readable knowledge on the Web. One of these approaches was the SHOE (Simple HTML Ontology Extensions) project, which took place at the University of Maryland [40]. The slogan for the SHOE project, which continues to be a popular quote in the Semantic Web community, was “*A little semantics goes a long way*,” and supporting this slogan the SHOE Base Ontology contained a very minimal set of concepts and relationships. Around this effort, a number of tools were created within the project including a semantic annotator for HTML pages and a semantic search tool.

Another early project was Ontobroker [18], which, like SHOE, looked at adding and using semantic annotations to HTML pages. These two early projects looked at what is now called Web ontology languages, and were driven less by the AI-inspired push for expressive languages, and more by the needs of the emerging Web – what would now be called semantic annotation or tagging.

Other early projects within Europe included On-To-knowledge (which began in January 2000) from which the SESAME repository was developed and also OIL [20] was set up as a cross-project initiative, merging an effort called XOL (XML Ontology Language) and the work emerging from Ontobroker. Approximately 18 months later the OntoWeb [70] network of excellence started, which was the birthplace for the Knowledge Web project [71].

In parallel with this Web representation work, W3C had begun to explore whether some sort of Web markup language could be defined to help bring data to the Web. The Metadata C Format working group was drafting a language that was later to be named the Resource Description Format (RDF). There was at this time a split between XML and RDE, which we do not have space here to recount but suffice to say that this added confusion to the overall story.

It is also worth noting here the dialogue that began in the late 1990s within the Knowledge Acquisition Workshop Series in Banff [72] on the relationship between knowledge acquisition, modeling, and the Web. One of the projects that came out of this discussion was IBROW³ [4], which examined how knowledge components could be reused through the Web. Elements of this project later influenced Semantic Web Services research (see [Semantic Web Services](#)).

1.5.1 Increasing Research Interest

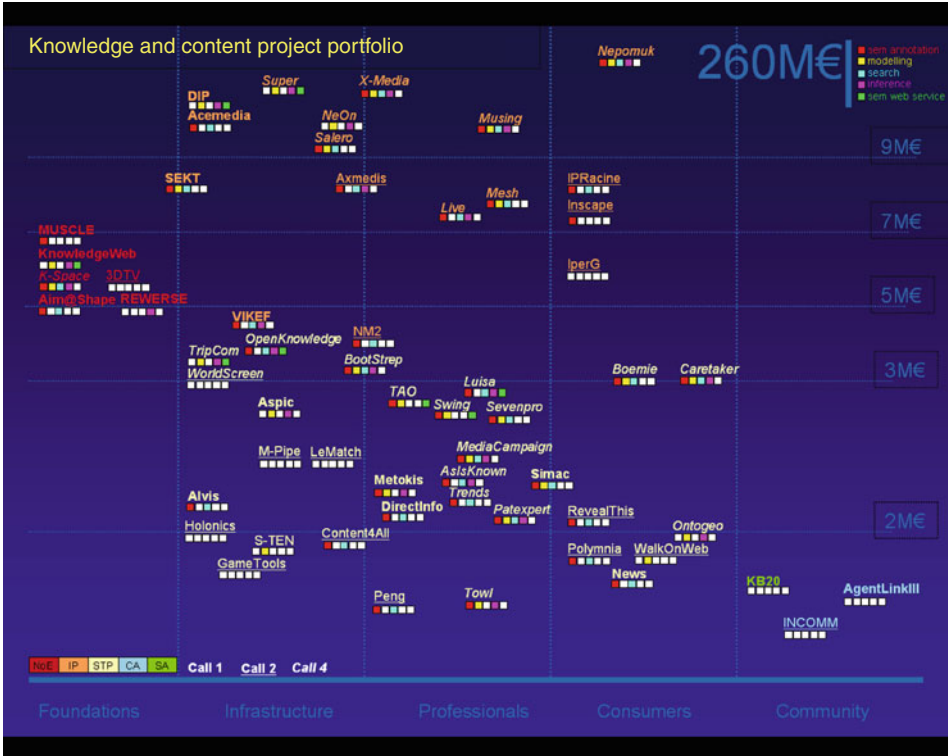
In 1999, one of the editors began a 3-year position as a funding agent for the US Defense Advanced Research Projects Agency and convinced them to invest in the technology. The primary argument was that Semantic Web technology could be used to help solve a lot of the Department of Defense's (and, of course, everyone else's) data integration problems. To help sell the US government on funding this research area, the techniques pioneered in Ontobroker and SHOE were used to build some demos showing the potential for these new languages.

Based on these demos, a project called the DARPA Agent Markup Language (DAML) was launched. MIT's Semantic Web Advanced Development, led by Sir Tim Berners-Lee, was funded under this program, with a proposal to base a language on top of RDF which was at the time being defined. RDF, like SHOE, used URIs to name concepts, an important aspect of "webizing" the representation languages for the Web. Along the way, the community (both research and industrial) came to accept Tim's name for this work: The Semantic Web.

In actuality, it is worth noting that the Semantic Web was a realization of part of Tim's original conception of the Web. In fact, in a 1994 talk (Web Conference, Geneva) he said:

- ▶ Documents on the web describe real objects and imaginary concepts, and give particular relationships between them. . . For example, a document might describe a person. The title document to a house describes a house and also the ownership relation with a person. . . . This means that machines, as well as people operating on the web of information, can do real things. For example, a program could search for a house and negotiate transfer of ownership of the house to a new owner. The land registry guarantees that the title actually represents reality.

As this work grew, it was decided that an effort was needed to bring together the key players in this emerging area. The outcome of this was a Dagstuhl Seminar held in 2000 [62]. The workshop was quite successful and led to a dramatic increase in funding especially in Europe. For example, [Fig. 1.3](#) below shows the snapshot of the projects funded by the



■ Fig. 1.3

A snapshot of the projects funded by the Knowledge and Content Unit in Luxembourg in 2005. This slide is available within a set at ftp://ftp.cordis.europa.eu/pub/ist/docs/kct/iswc05-slideshow_en.pdf (Figure used with the permission of the European Commission)

Knowledge and Content Unit in Luxembourg by type and funding, color coded according to the areas of semantic annotation, modeling, search, inference, and Semantic Web Services.

1.6 Related Resources

1.6.1 Semantic Web Events

1.6.1.1 Conferences

- **Asian Semantic Web Conference (ASWC)** – a Semantic Web conference series that targets the Asian continent. See <http://www.sti2.org/conferenceseries/asian-semantic-web-conferences> for details on the overall conference series.
- **European Semantic Technology Conference (ESTC)** – this conference tackles the commercial aspects for semantic technology with a European focus and is usually held

in Vienna, Austria. See <http://www.sti2.org/conferenceseries/european-semantic-technology-conferences> for details on the overall conference series.

- **Extended Semantic Web Conference (ESWC – formerly the European Semantic Web Conference)** – this annual conference had its seventh edition in 2010 and includes workshops and tutorials. The change in name relates to the conference series covering topics related to the application of semantics to mobile platforms, cloud computing, sensor networks, as well as the Web. See <http://www.sti2.org/conferenceseries/extended-semantic-web-conferences> for details on the overall conference series.
- **International Semantic Web Conference (ISWC)** – this annual conference is now (2010) in its ninth year and is a premier event for discussing Semantic Web topics. The event usually attracts around 600 participants and includes a research and in-use track as well as workshops and tutorials. See <http://iswc.semanticweb.org/> for details on the overall conference series.
- **I-Semantics** – is a European forum that examines semantics from a technological, economic, and social point of view. Details on the conference series can be found at <http://i-semantics.tugraz.at/>.
- **SemTech** – is an annual event that targets the professional area and the commercial deployment of semantic technology. This conference is usually held in San Francisco in the USA. Details on the 2010 event can be found at <http://semtech2010.semanticuniverse.com/>.
- **IEEE International Conference on Semantic Computing (ICSC)** – addresses the use of computational semantics to create, use, manage, and find content, where content refers to any type of resource including video, audio, text, processes, services, hardware, and networks. More details can be found at <http://www.ieee-icsc.org/>.
- **World Wide Web Conference** – this conference provides a forum for debate and discussion on the evolution of the Web, the standardization of the associated technologies, and the impact of the technologies on society and culture. This conference traditionally includes a Semantic Web track. More details can be found at <http://www.iw3c2.org/>.

1.6.1.2 Summer Schools and Tutorials

- **ESWC Summer School** – is a new Semantic Web summer school that will be held in conjunction with the ESWC conference. Details on the 2011 event can be found at <http://summerschool.eswc2011.org/>.
- **IEEE Summer School on Semantic Computing** – is a week-long event that up until now has been held on the Berkeley campus in California. See <http://www.sssc2010.org/> for details on the 2010 event.
- **Introduction to Semantic Web Tutorial** – has been held as a one-day event in conjunction with ISWC 2007, 2008, and 2010. See <http://people.csail.mit.edu/pcm/SemWebTutorial.html> for details on the 2010 event.

- **Summer School on Ontological Engineering and the Semantic Web** – this week-long summer school, which started in 2003, was initially funded by the EU OntoWeb and later the KnowledgeWeb project, and has always been held in Cercedilla, near Madrid, Spain. Details on the 2008 summer school can be found at <http://kmi.open.ac.uk/events/sssw08/>.

1.6.1.3 Semantic Web Journals and Magazines

- **Journal of Web Semantics** – this journal covers the main areas associated with the Semantic Web and publishes research, survey, ontology, and systems papers. More details on the journal can be found at http://www.elsevier.com/wps/find/journaldescription.cws_home/671322/description#description.
- **IEEE Intelligent Systems** – is a magazine that covers the broad area related to systems that act intelligently. It often includes papers though related to the Semantic Web. More details can be found at <http://www.computer.org/portal/web/intelligent/home>.
- **Applied Ontology** – covers conceptual modeling and ontology analysis. Details on the journal can be found at <http://www.iospress.nl/loadtop/load.php?isbn = 15705838>.
- **Semantic Web: Interoperability, Usability, Applicability** – is a Semantic Web journal that uses an open and transparent review process. Submitted manuscripts are posted on the journal's website to which researchers are free to post public reviews and authors to post responses. More details on the journal can be found at <http://www.semantic-web-journal.net/>.
- **International Journal On Semantic Web and Information Systems** – is a journal where aspects of the Semantic Web relevant to the Computer Science and Information Systems communities are discussed. See <http://www.ijswis.org/> for more details.

1.6.1.4 Semantic Websites

- <http://www.iswsa.org/> – the Web page for the Semantic Web Science Association that runs ISWC
- <http://semanticweb.org/> – a Wiki page for the Semantic Web community
- <http://www.sti2.org/> – contains a list of resources including events that are organized by STI International, a networked organization for parties interested in Semantic Technology
- <http://www.w3.org/2001/sw/> – the W3C page that lists W3C Semantic Web activities
- <http://www.linkeddata.org> – provides a home for and pointers to resources associated with the linked data initiative
- <http://data.semanticweb.org/> – the Semantic Web Conference Corpus also known as the Semantic Web Dog Food Corpus, which contains data and ontologies related to Semantic Web events (including ESWC, ISWC, and WWW mentioned above) and researchers, organizations, and papers related to the area

1.6.1.5 Sources Introducing the Semantic Web

A number of videos and websites exist that outline the basic notions behind the Semantic Web.

- http://videlectures.net/iswc08_hendler_ittsw/ – the Introduction to the Semantic Web Tutorial from ISWC 2008
- <http://www.youtube.com/watch?v=OGg8A2zfWKg> – a very clear introduction to the Semantic Web from Digital Bazaar Inc.
- <http://infomesh.net/2001/swintro/#whatIsSw> – a simple and comprehensive introduction for anyone trying to understand the Semantic Web

1.6.1.6 Books

- **A Semantic Web Primer** (2nd Edition) Grigoris Antoniou and Frank van Harmelen (MIT Press) – a textbook suitable for undergraduates which gives a broad introduction to the motivation behind the Semantic Web, as well as its applications and supporting technologies. The book introduces the specific languages associated with the Semantic Web including RDF and OWL. Additional material including slides can be found at <http://www.semanticwebprimer.org/>.
- **Foundations of Semantic Web Technologies** by Pascal Hitzler, Markus Krötzsch, and Sebastian Rudolph (Chapman and Hall) – this book covers RDF Schema, OWL, rules, and query languages, such as SPARQL. Recent developments such as OWL 2 and RIF are also covered.
- **Semantic Web for Dummies** by Jeffrey T. Pollock (Wiley Inc.) – provides a gentle introduction to the Semantic Web covering the area as a set of technologies, a social phenomena, and a web-scale architecture.
- **The Semantic Web: Semantics for Data and Services on the Web** by Vipul Kashyap, Christoph Bussler, and Matthew Moran (Springer) – covers the Semantic Web from a data and process perspective and includes basic coverage of XML, RDF, and ontologies.
- **Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL** by Dean Allemang and James Hendler (Morgan Kaufmann) – is a practical book aimed at practitioners who wish to create semantic models using Semantic Web technologies.

1.7 Selected Successes in the Commercial Sphere

During the just over decade covering the Semantic Web as a research topic, one of the most common criticisms was that the work would never be commercially successful due to problems with the scalability and usability of semantic technology. The debate on whether Semantic Web technologies will be commercially successful is now over and has been replaced instead with a discussion on what specific forms deployed commercial semantic

applications will take. Moreover, a number of commercial announcements have been made recently, which indicate that one is moving from an early adopters phase to more mainstream markets for semantic technologies. A longer discussion of this can be found in [▶ *Semantic Technology Adoption: A Business Perspective*](#).

1.7.1 Oracle

Oracle's support for semantic technology started with its 10gR2 system and a number of enhancements were made when 11g was subsequently released. On their website [77], Oracle now state that 11g supports a number of core technologies including RDF(S), OWL, SKOS, and SPARQL. Also, support is provided for a number of open-source tools, including Jena, Sesame, and Protégé, and a number of third-party entity extraction services, such as OpenCalais and GATE.

A technical perspective on 11g including benchmarking information can be found in [▶ *Storing the Semantic Web: Repositories*](#). However, of interest here is the fact that a mainstream conventional IT provider is now an advocate of the Semantic Web. One of the main reasons for this is that as with many commercial shifts, this was a requirement from Oracle customers, particularly in the areas of pharmaceuticals, life sciences, and health care, who need to integrate large amounts of data from many different sources. This type of data integration at scale and across many heterogeneous sources which cannot be changed is one where semantic repositories cope well. Additionally, in these areas, reasoning capabilities are useful in supporting the mining and analysis of the data.

1.7.2 Facebook's Open Graph Protocol

In May 2010, Facebook announced their Open Graph Protocol [63], which is based on RDFa. The exact relationship between Open Graph and RDFa is discussed in [▶ *Semantic Annotation and Retrieval: Web of Hypertext – RDFa and Microformats*](#) in detail. Here, the focus is on the impact of the announcement. In short, the Open Graph protocol facilitates the integration of Web resources into a Facebook social graph. A Facebook “like” button can be embedded in any Web page allowing Facebook users to “like” any Web resource. [▶ *Figure 1.4*](#) below shows this facility in use in an Open University news system enabling readers to express preferences over published stories. It is seen in the figure that three readers have expressed that they like the story. These preferences also allow site owners to track the demographic data of users visiting their site.

In the last few months, a number of commercial companies have built sites around this feature. Levi's have a dedicated store, which incorporates a like button for every product [79]. Also, Amazon have integrated their recommendation system to use Facebook profiles through Open Graph. Facebook have also recently integrated Open Graph into the Facebook SDK for the iPhone and Android platforms.

There are two main reasons for highlighting this deployment of semantic technology. Firstly, now in effect there are 500 million (and currently growing) Facebook users



■ Fig. 1.4

A screen snapshot from the online news system of the Knowledge Media Institute where Facebook users can say that they like a story

semantically annotating the Web from fixed and mobile devices. The probability is that this will in the short to medium term be a major source for semantic data. When making the announcement, Facebook’s CEO, Mark Zuckerberg, claimed that the technology would result in over one billion like buttons spreading across the Web in the first 24 h [81].

The second more general aspect about the announcement is that one of the world’s largest Web companies deems Semantic Web technology a suitable choice on which to center its corporate strategy. In particular, Facebook currently claim that Open Graph is “the most transformative thing we’ve ever done for the web” [83] – which is a very strong endorsement for semantic technology.

1.7.3 Google Buys Metaweb

In July 2010, Google bought Metaweb, the company which maintains Freebase. As reported in [Semantic Annotation and Retrieval: Web of Data](#), Freebase is a major source of cross-domain data within the growing linked dataset. Currently, Freebase has around 12 million items including movies, books, and organizations. According to Google’s

Director of Product Management, Freebase will enable the company to target more complex questions such as “actors over 40 who have won at least one Oscar?” [69]. From a linked data viewpoint, one interesting aspect of this purchase is that Google intends to maintain Freebase as a free and open resource. This announcement builds upon Google’s use of microformats and RDFa to power their Rich Snippets feature, which is used to enhance returned search results.

1.7.4 BBC Football World Cup 2010 Website

For the 2010 Football World Cup, the BBC website used a semantic-based publishing framework based on an RDF triple store described in [▶ Storing the Semantic Web: Repositories](#). The Website included over 700 pages describing the 32 teams, 8 groups, and the associated hundreds of footballers that took part in the event. The Web pages were dynamically aggregated using a football ontology describing concepts associated with the World Cup (e.g., teams, players, and groups) as well as publication assets (e.g., story, blog, image, and video).

One can see the page describing the England midfielder Frank Lampard. Using the underlying ontology and the stored RDF data, the page shows the basic statistics for Frank’s performance in the World Cup: the number of games played, the number of goals scored and goal assists, the number of shots on and off target for the goal, and also statistics related to discipline, such as yellow and red cards, and the number of fouls committed by and committed on Frank. The key advantage here of course is that the page is generated dynamically from the data and, thus, the publication process is streamlined and maintenance effort is drastically reduced (see http://news.bbc.co.uk/sport1/hi/football/world_cup_2010/groups_and_teams/team/england/frank_lampard).

The use of semantic technology was deemed to be successful and the website proved popular dealing with several million page requests every day throughout the World Cup. BBC now plans to use the technology again for the London Olympics in 2012 and the Chief Technical Architect, Journalism and Knowledge, BBC Future Media and Technology stated: “We look forward to seeing the use of Linked Data grow as we move towards a more Semantic Web” [58].

Technical details on the above can be found in [▶ Storing the Semantic Web: Repositories](#).

1.7.5 Apple Buys SIRI

Siri is a free iPhone App, currently only available in the USA, which acts as a virtual personal assistant for a set of common tasks. [▶ Figure 1.5](#) shows the main interface for Siri. User requests, which can be typed or spoken, are given through a dialog interface customized for smart phone screens. Context information, including the user’s location and personal preferences, the time, and the selected task, are used to aid in understanding



■ Fig. 1.5

A screen snapshot of the SIRI interface [27] showing the interface for the iPhone (Image courtesy: Tom Gruber, © Siri)

the posed request. The currently supported tasks include booking a table at a restaurant, for a movie, or for an event, and requesting a local taxi or finding local businesses.

In [Fig. 1.6](#), the role that semantics plays within the overall architecture can be seen. In addition to the sophisticated dialog system, domain and task models are used to support the combining of online services to fulfill the requested task. There is in fact

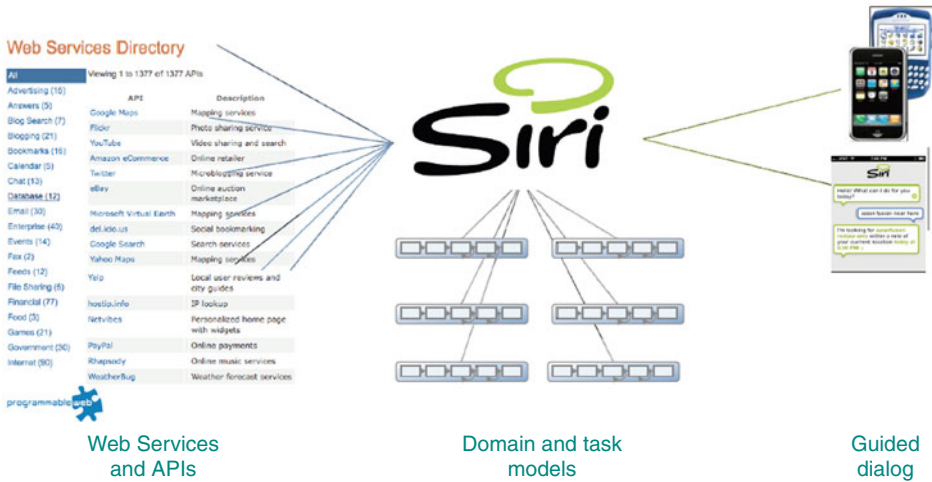


Fig. 1.6

The Siri overall architecture [26] where domain and task models are used to combine online services and Web APIs to satisfy user requests given from a mobile device (Image courtesy: Tom Gruber, © Siri)

a (partly historical) commonality between the approach that Siri takes to combining services and the WSMO [21] approach highlighted in [Semantic Web Services](#). One of the main functionalities provided by semantics in the Siri architecture is in providing the mapping between a task “book a table for 2 people at a Mexican restaurant in the local vicinity” and online services (restaurant finding services, recommendation services, restaurant table booking services, etc.).

The main benefit that Siri provides for the end user is that a simple conversation replaces the effort of combining either a sequence of Web searches or a sequence of mobile phone App interactions. Siri had raised approximately \$24 million in venture funding and was bought by Apple in the summer of 2010 for an estimated value of between \$100 and \$200 million [59].

It can be seen above that semantic technology is beginning to enter the mainstream. Also, by and large, it is the simpler technologies which are data-centric that have been taken up. There are a number of views that one could take on this. One is that it should be expected that by their very nature, real-world Web applications will be dominated by data rather than conceptual structures. Second, even with the successes emerging now, the Semantic Web is still in a preliminary phase of commercialization and it will take time to progress to Web applications, which require more complex conceptual reasoning.

The acquisition of Siri runs somewhat counter to the reasoning above and indicates that there may be space for more complex forms of reasoning, as is required to deal with services and Web APIs.

1.8 Future

Chapter on [▶ Future Trends](#) contains predictions of semantic technologies 5, 10, and 15 years into the future from application and core technology points of view. Reflecting on the last decade of research into the Semantic Web, two issues seem clear. Firstly, as outlined above, at this point semantic technology is becoming mainstream and we will continue to see deployment of semantics in the commercial sector. It is envisaged that in the near term, organizations will make significant portions of their data available on the Web using semantic technologies. Moreover, the emergence of data will grow in a way analogous to the way in which the Web grew. At the beginning of the Web, it was often asked what would motivate individuals and organizations to put resources into creating and developing websites. Over the history of the Web, we have seen a progressive escalation in this effort. Corporations will now have entire departments dedicated to maintaining their presence on the Web. Web presence is seen as a requirement rather than a luxury, and the Google ranking of an organization can determine its success. As a first step toward the vision outlined in the *Scientific American* paper [6], a semantic data presence will soon become a requirement rather than a luxury. When advocating that semantic technology would be a core pillar of the UK's Digital Britain initiative, Gordon Brown (when he was the UK Prime Minister) declared one significant benefit would be the reduction in the cost of maintaining government websites [73]. Thus, linked data moves the effort of creating and maintaining websites and Web applications over organizational data to external parties. Chapter [▶ Knowledge Management in Large Organizations](#) discusses related issues from an enterprise perspective.

Secondly, the Web is changing in a number of ways. As covered in [▶ Social Semantic Web](#) and mentioned briefly above, there is already a link between social networking sites and the Semantic Web. It is expected to see a growth in platforms for Web applications based upon combinations of social networking and semantic technologies, harnessing the power of human networks and automated reasoning. A discussion is currently taking place related to which forces will dominate the way the Internet is used. Wired recently ran an article with the title “The Web Is Dead. Long Live the Internet” [84]. In this article, the authors saw three trends emerging. Firstly, that video and peer-to-peer network traffic are beginning to take a large proportion of Internet traffic when compared to pure Web communication. Secondly, that as predicted in several places, the number of users accessing the Internet from mobile devices will soon surpass the number who access it from PCs. A consequence of the shift to mobile devices such as the iPhone and iPad is that specialist Apps designed for a single purpose will be used more than general-purpose Web browsers. A third trend from the commercial perspective is that the Internet will be dominated by a relatively small number of large players, such as Apple, who will act like the media empires of the third quarter of the twentieth century. In the article “Google: The search party is over” [80], an analysis of the differences between the stock prices and values of Google and Apple (\$156 vs \$236 billion, respectively) is used to support a claim that search will no longer be the most significant part of Web applications. An associated claim

is that search will be supplanted by information gathering from colleagues and friends via social networking sites.

These claims are not agreed by all however. For example, in a TechCrunch article “When Wrong, Call Yourself Prescient Instead” [82], the authors cite previous predictions of the Web’s demise which proved to be false. One thing that can be assumed safely is that the debate will continue for some time. After a decade of research and as shown in the rest of this book, the Web is a global infrastructure that benefits significantly from the use of semantics. Semantics supports a broad range of tasks including data sharing and data integration at scale, knowledge management, decision making, data analysis, search, and the use and management of Web applications based on Web APIs and services, as well as a variety of vertical sectors such as government, science, business, and media. Given the success thus far, it is clear that semantic technology will also play a major role in other global network infrastructures based on, for example, mobile devices and sensor nets. Whatever form future planet-scale networks take, it has certainly been an exhilarating journey so far and we look forward to the next decade.

Acknowledgments

We thank Ian Horrocks and Michael Kifer from preventing mistakes in the sections of the chapter related to logic. We also thank Neil Benn for his help in the final formatting stages.

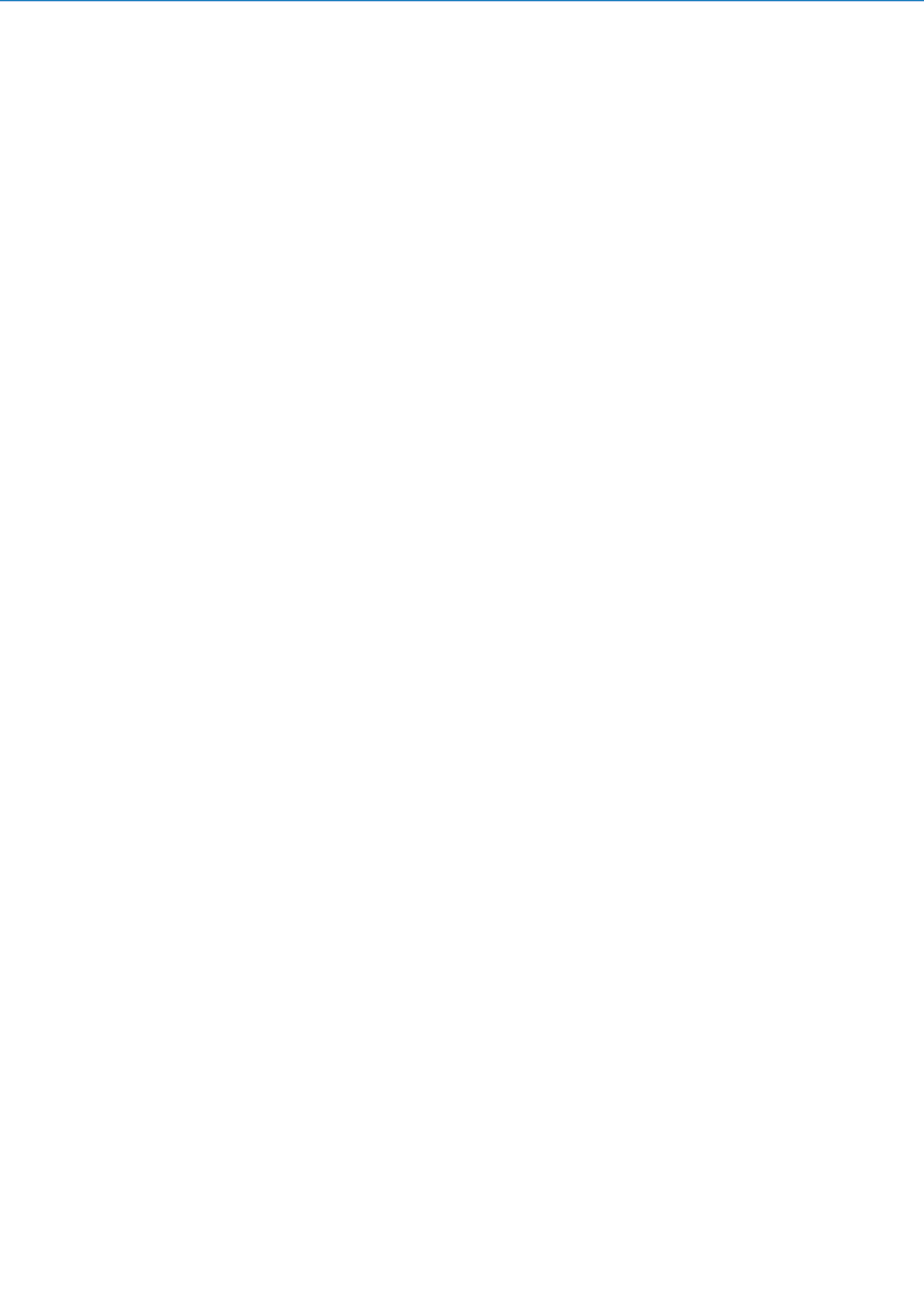
References

1. Adida, B., Birbeck, M. (eds.): RDFa primer: bridging the human and data webs, W3C Working Group Note (Oct 2008)
2. Antoniou, G., van Harmelen, F.: A Semantic Web Primer, 2nd edn. MIT Press, Cambridge (2008)
3. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook. Cambridge University Press, Cambridge (2003)
4. Benjamins, V.R., Plaza, E., Motta, E., Fensel, D., Studer, R., Wielinga, B., Schreiber, G., Zdrahal, Z., Decker, S.: IBROW3 an intelligent brokering service for knowledge-component reuse on the world-wide web. In: Proceedings of the 11th Banff Knowledge Acquisition for Knowledge-Based System Workshop (KAW 1998), Banff. <http://ksi.cpsc.ucalgary.ca/KAW/KAW98/benjamins3/> (1998). Accessed Aug 2010
5. Berners-Lee, T.: Information management: a proposal. March 1989 and later redistributed unchanged apart from the date added in May 1990. <http://www.w3.org/History/1989/proposal.html> (1989). Accessed Aug 2010
6. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. *Scientific American Magazine*, pp. 29–37 (May 2001)
7. Bizer, C., Heath, T., Berners-Lee, T.: Linked data – the story so far. *Int. J. Semant. Web Inf. Syst.* 5(3), 1–22 (2009)
8. Brachman, R.J., Levesque, H.J.: Knowledge Representation and Reasoning. Morgan Kaufmann, San Francisco (2004)
9. Brickley, D., Guha, R.V. (eds.): RDF vocabulary description language 1.0: RDF schema. W3C Recommendation (Feb 2004)
10. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.* 30(1–7), 107–117 (1998)
11. Bush, V.: As we may think. *The Atlantic Monthly*, July 1945. <http://www.theatlantic.com/past/docs/unbound/flashbks/computer/bushf.htm> (1945). Accessed Aug 2010

12. Cerf, V., Kahn, B.: "A Protocol for Packet Network Interconnection", which specified in detail the design of a Transmission Control Protocol (TCP) (1974)
13. Chen, W., Kiefer, M., David, S.W.: HiLog: a foundation for higher-order logic programming. *J. Log Program* 15(3), 187–230 (1993)
14. Clocksin, W.F., Mellish, C.S.: *Programming in Prolog*, 5th edn. Springer, New York (2003)
15. Codd, E.F.: *The Relational Model for Database Management: Version 2*. Addison-Wesley Longman, New York (1990)
16. Dean, M., Schreiber, G. (eds.): *OWL web ontology language reference*, W3C Recommendation (Feb 2004)
17. Feigenbaum, E.A.: *The art of artificial intelligence: themes and case studies of knowledge engineering*. In: *Proceedings of the Fifth International Joint Conference on Artificial Intelligence (IJCAI 1977)*, Cambridge (1977)
18. Fensel, D., Decker, S., Erdmann, M., Studer, R.: *Ontobroker: the very high idea*. In: *Proceedings of the 11th International Florida Artificial Intelligence Research Society Conference (FLAIRS 1998)*, Sanibel Island, pp. 131–135 (1998)
19. Fensel, D., Angele, J., Decker, S., Erdmann, M., Schnurr, H.-P., Studer, R., Witt, A.: *Lessons learned from applying AI to the web*. *J. Cooperat. Inf. Syst.* 9(4) (2000)
20. Fensel, D., van Harmelen, F., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F.: *OIL: an ontology infrastructure for the semantic web*. *IEEE Intell. Syst.* 16(2), 38–45 (2001)
21. Fensel, D., Lausen, H., Polleres, A., De Bruijn, J., Stollberg, M., Roman, D., Domingue, J.: *Enabling Semantic Web Services: The Web Service Modeling Ontology*. Springer, New York (2007)
22. Fensel, D.: *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce*. Springer, Berlin (2001), 2nd edn. Springer (2003)
23. Garcia-Molina, H., Ullman, J.D., Widom, J.: *Database Systems: The Complete Book*, 2nd edn. Prentice Hall, New Jersey (2009)
24. Giarratano, J.C., Riley, G.D.: *Expert Systems: Principles and Programming*, 4th edn. PWS, Boston (2004)
25. Gruber, T.R.: *A translation approach to portable ontology specifications*. *Knowl. Acquis.* 5(2), 199–220 (1993)
26. Gruber, T.: *Siri: a virtual personal assistant*. Keynote Presentation at Semantic Technologies Conference. <http://tomgruber.org/writing/semtech09.htm> (2009). Accessed Aug 2010
27. Gruber, T.: *Big Think Small Screen: how semantic computing in the cloud will revolutionize the consumer experience on the phone*. Keynote Presentation at Web 3.0 Conference. <http://tomgruber.org/writing/web30jan2010.htm> (2010). Accessed Aug 2010
28. Halpin, H., Davis, I. (eds.): *GRDDL primer*, W3C Working Group Note (June 2007)
29. Hedman, S.: *A First Course in Logic*. Oxford University Press, Oxford (2004)
30. Horrocks, I.: *Using an expressive description logic: FaCT or fiction?* In: *Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR' 1998)*, pp. 636–647 (1999)
31. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosz, B., Dean, M.: *SWRL: A semantic web rule language combining OWL and RuleML*, W3C Member Submission (May 2004)
32. ter Horst, H.J.: *Completeness, decidability and complexity of entailment for RDF schema and a semantic extension involving the owl vocabulary*. *J. Web Semant.* 3(2–3), 79–115 (2005)
33. Isaac, A., Summers, E. (eds.): *SKOS simple knowledge organization system primer*, W3C Working Group Note (Aug 2009)
34. Jurafsky, D., Martin, J.H.: *Speech and Language Processing*, 2nd edn. Prentice Hall, New Jersey (2009)
35. Kelly, J.: *The Essence of Logic*. Prentice Hall, New Jersey (1997)
36. Kifer, M., Boley, H. (eds.): *RIF overview*, W3C Working Group Note (June 2010)
37. Kifer, M., Lausen, G., Wu, J.: *Logic foundations of object-oriented and frame-based systems*. *J. ACM* 42, 741–843 (1995)
38. Lausen, H., Farrell, J. (eds.): *Semantic annotations for WSDL and XML schema*, W3C Recommendation (Aug 2007)
39. Lloyd, J.W.: *Foundations of Logic Programming*, 2nd edn. Springer, Berlin (1987)
40. Luke, S., Specto, L., Rager, D., Hendler, J.: *Ontology-based Web agents*. In: *Proceedings of the First International Conference on Autonomous Agents (ICAA 1997)*, Marina del Rey, pp. 59–66 (1997)

41. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
42. Manola, F., Miller, E. (eds.): RDF primer, W3C Recommendation (Feb 2004)
43. Maslow, H.: The Psychology of Science. Harper & Row, New York (1966)
44. Mauritius National Assembly: The constitution. <http://www.gov.mu/portal/AssemblySite/menuitem.ee3d58b2c32c60451251701065c521ca/>. Accessed 6 Sept 2010
45. Mead, G.H.: Mind, Self, and Society. The University of Chicago Press, Chicago (1934)
46. Moens, M.-F.: Information Extraction: Algorithms and Prospects in a Retrieval Context. Springer, New York (2006)
47. Motik, B., Grau, B.C., Horrocks, I., Wu, Z., Lutz, C. (eds.): OWL 2 web ontology language profiles, W3C Recommendation (Oct 2009)
48. Nelson, T.H.: A file structure for the complex, the changing, and the indeterminate. In: Proceedings of the 20th National Conference, Association for Computing Machinery, New York (1965)
49. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL web ontology language semantics and abstract syntax. Section 5. RDF-compatible model-theoretic semantics, W3C (2004)
50. Pingdom: Internet 2009 in numbers. <http://royal.pingdom.com/2010/01/22/internet-2009-in-numbers/> (2010). Accessed 6 Sept 2010
51. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF, W3C Recommendation (Jan 2008)
52. Reynolds, D.: OWL 2 RL in RIF, W3C Working Group Note (June 2010)
53. Robinson, A., Voronkov, A. (eds.): Handbook of Automated Reasoning. Elsevier Science, Amsterdam (2001)
54. Russell, S., Norvig, P.: Artificial Intelligence – A Modern Approach, 2nd edn. Prentice Hall, New Jersey (2003)
55. Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., Van de Velde, W., Wielinga, B.: Knowledge Engineering and Management: The Common KADS Methodology. MIT Press, Cambridge (2000)
56. Tomasello, M.: Origins of Human Communication. MIT Press, Cambridge (2008)
57. http://www.bbc.co.uk/blogs/bbcinternet/2010/07/bbc_world_cup_2010_dynamic_sem.html. Accessed 6 Sept 2010
58. http://www.bbc.co.uk/blogs/bbcinternet/2010/07/the_world_cup_and_a_call_to_ac.html. Accessed 6 Sept 2010
59. <http://www.businessinsider.com/apple-buys-siri-a-mobile-assistant-app-as-war-with-google-heats-up-2010-4>. Accessed 6 Sept 2010
60. <http://challenge.semanticweb.org/>. Accessed 6 Sept 2010
61. http://www.comscore.com/Press_Events/Press_Releases/2010/1/Global_Search_Market_Grows_46_Percent_in_2009. Accessed 6 Sept 2010
62. <http://www.dagstuhl.de/en/program/calendar/semhp/?semnr=00121>. Accessed 6 Sept 2010
63. <http://developers.facebook.com/docs/opengraph>. Accessed 6 Sept 2010
64. <http://dictionary.reference.com/browse/meaning>. Accessed 6 Sept 2010
65. <http://dublincore.org/>. Accessed 6 Sept 2010
66. <http://www.everyhit.com>. Accessed 6 Sept 2010
67. <http://www.fipa.org/>. Accessed 6 Sept 2010
68. <http://www.foaf-project.org/>. Accessed 6 Sept 2010
69. <http://googleblog.blogspot.com/2010/07/deeper-understanding-with-metaweb.html>. Accessed 6 Sept 2010
70. <http://www.ist-world.org/ProjectDetails.aspx?ProjectId=e132f5b74a41456f95611eb7ad3abfd3>. Accessed 6 Sept 2010
71. <http://knowledgeweb.semanticweb.org/>. Accessed 6 Sept 2010
72. <http://ksi.cpsc.ucalgary.ca/KAW/>. Accessed 6 Sept 2010
73. <http://www2.labour.org.uk/gordon-browns-speech-on-building-britains-digital-future,2010-03-26>. Accessed 6 Sept 2010
74. <http://www.larkc.eu/>. Accessed 6 Sept 2010
75. <http://microformats.org/>. Accessed 6 Sept 2010
76. <http://www.openlinksw.com/weblog/oerling/?id=1614>. Accessed 6 Sept 2010
77. <http://www.oracle.com/technetwork/database/options/semantic-tech/index.html>. Accessed 6 Sept 2010
78. <http://www.pandia.com/sew/383-web-size.html>. Accessed 6 Sept 2010
79. <http://store.levi.com/>. Accessed 6 Sept 2010

80. <http://tech.fortune.cnn.com/2010/07/29/google-the-search-party-is-over/>. Accessed 6 Sept 2010
81. <http://techcrunch.com/2010/04/21/facebook-like-button/>. Accessed 6 Sept 2010
82. <http://techcrunch.com/2010/08/17/when-wrong-call-yourself-prescient-instead/>. Accessed 6 Sept 2010
83. http://technology.timesonline.co.uk/tol/news/tech_and_web/the_web/article7104354.ece. Accessed 6 Sept 2010
84. http://www.wired.com/magazine/2010/08/ff_webrip/all/1
85. <http://WordNet.princeton.edu/>. Accessed 6 Sept 2010
86. Wikipedia: Controlled vocabulary. http://en.wikipedia.org/wiki/Controlled_vocabulary (2010). Accessed 6 Sept 2010
87. Wikipedia: Energy. <http://en.wikipedia.org/wiki/Energy> (2010). Accessed 6 Sept 2010
88. Wikipedia: Equipment. <http://en.wikipedia.org/wiki/Equipment> (2010). Accessed 6 Sept 2010
89. Wikipedia: Formal semantics. http://en.wikipedia.org/wiki/Formal_semantics (2010). Accessed 6 Sept 2010
90. Wikipedia: Idea. <http://en.wikipedia.org/wiki/Idea> (2010). Accessed 6 Sept 2010
91. Wikipedia: Intention. <http://en.wikipedia.org/wiki/Intention> (2010). Accessed 6 Sept 2010
92. Wikipedia: Machine. <http://en.wikipedia.org/wiki/Machine> (2010). Accessed 6 Sept 2010
93. Wikipedia: NLS (computer system). [http://en.wikipedia.org/wiki/NLS_\(computer_system\)](http://en.wikipedia.org/wiki/NLS_(computer_system)) (2010). Accessed 6 Sept 2010
94. Wikipedia: OSI model. http://en.wikipedia.org/wiki/OSI_model (2010). Accessed 6 Sept 2010
95. Wikipedia: Purpose. <http://en.wikipedia.org/wiki/Purpose> (2010). Accessed 6 Sept 2010
96. Wikipedia: Second-order logic. http://en.wikipedia.org/wiki/Second-order_logic (2010). Accessed 6 Sept 2010
97. Wikipedia: Semantic HTML. http://en.wikipedia.org/wiki/Semantic_HTML (2010). Accessed 6 Sept 2010
98. Wikipedia: Semantics. <http://en.wikipedia.org/wiki/Semantics> (2010). Accessed 6 Sept 2010
99. Wikipedia: SLD resolution. http://en.wikipedia.org/wiki/SLD_resolution (2010). Accessed 6 Sept 2010
100. Wikipedia: SQL. <http://en.wikipedia.org/wiki/SQL> (2010). Accessed 6 Sept 2010
101. Wikipedia: Tag cloud. http://en.wikipedia.org/wiki/Tag_cloud (2010). Accessed 6 Sept 2010
102. Wikipedia: Taxonomies. <http://en.wikipedia.org/wiki/Taxonomies> (2010). Accessed 6 Sept 2010
103. Wikipedia: Tower of Babel. http://en.wikipedia.org/wiki/Tower_of_Babel (2010). Accessed 6 Sept 2010
104. Wiktionary: Device. <http://en.wiktionary.org/wiki/device> (2010). Accessed 6 Sept 2010
105. World Wide Web Consortium: OWL web ontology language reference, W3C Recommendation. <http://www.w3.org/TR/owl-ref/> (Feb 2004). Accessed 6 Sept 2010
106. World Wide Web Consortium: RDB2RDF working group. <http://www.w3.org/2001/sw/rdb2rdf/>. Accessed 6 Sept 2010
107. World Wide Web Consortium: RDF primer, W3C Recommendation. <http://www.w3.org/TR/rdf-primer/>. Accessed 6 Sept 2010
108. World Wide Web Consortium: The global structure of an HTML document, W3C Recommendation. <http://www.w3.org/TR/html401/struct/global.html#edef-META>. Accessed 6 Sept 2010
109. World Wide Web Consortium: XML technology, W3C Standard. <http://www.w3.org/standards/xml/>. Accessed 6 Sept 2010
110. Yeates, G.: Earthworms. Te Ara – the encyclopedia of New Zealand (updated 1 March 2009). <http://www.teara.govt.nz/en/earthworms/3/1> (2009). Accessed 6 Sept 2010



2 Semantic Web Architecture

Andreas Harth¹ · Maciej Janik² · Steffen Staab²

¹AIFB Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

²University of Koblenz-Landau, Koblenz, Germany

2.1	<i>A Semantic Web Scenario from Today</i>	44
2.2	<i>Requirements for a Semantic Web Architecture</i>	45
2.2.1	Web Architecture	45
2.2.2	Requirements for an Architecture for the Web of Data	46
2.3	<i>First Principles for a Semantic Web Architecture</i>	49
2.3.1	Protocols and Languages	50
2.3.2	Software Components	51
2.3.3	System Architecture Styles	53
2.4	<i>Building Blocks</i>	54
2.4.1	Referencing, Transport Protocol, and Linked Data Principles	54
2.4.2	Data Interchange	55
2.4.3	Querying, Updating, and Views	56
2.4.4	Ontology and Reasoning	61
2.4.5	Rules and Rule Engines	62
2.4.6	Security and Encryption	63
2.4.7	Identity Management and Alignment	64
2.4.8	Provenance and Trust	65
2.4.9	User Interaction and Visualization	67
2.5	<i>Related Resources</i>	69
2.6	<i>Future Issues</i>	70
2.7	<i>Cross-References</i>	71

Abstract: The Semantic Web extends the existing Web, adding a multitude of language standards and software components to give humans and machines direct access to data. The chapter starts with deriving the architecture of the Semantic Web as a whole from first principles, followed by a presentation of Web standards underpinning the Semantic Web that are used for data publishing, querying, and reasoning. Further, the chapter identifies functional software components required to implement capabilities and behavior in applications that publish and consume Semantic Web content.

One of the key goals of Semantic Web technologies is to provide machines with a more sapient understanding of data. To this end, an increasing number of websites publish data in standards defined by the World Wide Web Consortium (W3C). Given a wider availability of quality data online, applications can leverage a common data access and integration layer for providing elaborate services to users. The chapter derives the architecture of the Semantic Web from first principles, gives an overview of the architecture of Semantic Web applications, and covers building blocks of the Semantic Web in more detail.

The chapter is structured as follows: 🔗 [Sect. 2.1](#) introduces a scenario describing an information need which is difficult to satisfy using traditional Web technologies based on hypertext, but will be easy to answer using Semantic Web technologies. 🔗 [Section 2.2](#) presents detailed requirements for an architecture. 🔗 [Section 2.3](#) derives an architecture from first principles. 🔗 [Section 2.4](#) covers the individual components deployed on the Semantic Web. 🔗 [Section 2.5](#) lists related resources. 🔗 [Section 2.6](#) concludes.

2.1 A Semantic Web Scenario from Today

“Which type of music is played in UK radio stations?” and “Which radio station is playing titles by Swedish composers?” are the types of questions that are very hard to answer using existing Web search engines; the upcoming Semantic Web provides a better framework to facilitate answering such queries. The information required to answer these questions is available on the Web. In fact, a large amount of such information already exists in formats amenable to machine processing on the Semantic Web. The reason that Web search engines fail at answering such questions is that they are limited to analyzing Web content – mostly documents in natural language – one page at a time, while the Semantic Web allows for combining data that are distributed across many different sources and described in a machine-interpretable manner.

For example, how may one pursue answering the questions related to playlists of UK radio stations? Playlists of BBC radio shows are published online in Semantic Web formats. A music group such as “ABBA” has an identifier (<http://www.bbc.co.uk/music/artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist>) that may be used to relate the music group to information at Musicbrainz (<http://musicbrainz.org/>), a music community portal exposing data on the Semantic Web [65]. MusicBrainz knows about band members, such as Benny Andersson, and about genre of artists and songs. In addition,

MusicBrainz aligns its information with Wikipedia, for example, to be able to include the biography of an artist, or to add facts from DBpedia [2], a version of Wikipedia in Semantic Web formats. Information on UK radio stations may be found in lists on Web pages such as <http://www.listenlive.eu/uk.html>, which can be translated to a similar Semantic Web representation – descriptions of things and their relationships.

The meaning of such relationships are explained online, too, using a set of ontologies available on the Web, such as Dublin Core, for describing general properties of information resources, SKOS for covering taxonomic descriptions, and specialized ontologies covering the music domain. Data at the BBC currently use at least nine different ontologies, with varying degrees of formality (<http://www.bbc.co.uk/ontologies/programmes>).

Given the available data, one may answer questions such as the frequency of certain music genres played on UK radio stations, radio stations playing Swedish composers, and many many more. However, having access to and leveraging such data does not come for free. The outlined scenario and likewise other use cases require generic software components, languages, and protocols that must interact in a seamless manner to be able to satisfy such requests. The chapter investigates the construction of the required infrastructure at large, that is, the Semantic Web architecture, and analyzes the requirements that come from the technical need to identify and relate data, and the organizational needs to maintain the Semantic Web as a whole – even if single components shake or break.

2.2 Requirements for a Semantic Web Architecture

The following section develops functional requirements for a Semantic Web architecture from the scenario given above (cf. [Sect. 2.2.2](#)). Achieving such functional capabilities requires an unprecedented growth of openly available data covering a wide range of domains and involving large amounts of people and organizations. Such phenomenal and fast growth is a nonfunctional (i.e., not a purely technological) requirement that has only been achieved by the World Wide Web. Thus, the first consideration is the architecture of the Web, to be able to learn from its design considerations, and to derive additional nonfunctional requirements later on.

2.2.1 Web Architecture

The World Wide Web is the largest software system known today. Its explosive growth is tightly associated with its underlying software architecture. Important for the Web's success was that:

- Many people could set up a Web server easily and independently from each other.
- More people could create documents, put them online, and link them to each other.
- Even more people could use a browser and access any Web server (and actually some other ones, like FTP or gopher servers) for retrieving these documents.

The fact that each individual system has been coupled only very loosely with the other one and that document creation, document delivery, and document browsing could happen in isolation in each of the many individual nodes was of key importance for enabling the fast adoption of the early Web. In this way, the Web architecture allowed for a graceful degradation of user experience when:

- The network was partially slow (“World Wide Wait”), while other parts might still operate at full speed
- Single Web servers broke, because others would still work
- Hyperlinks were broken, because others would still lead you somewhere

Thus, the Web’s architecture allowed for fast growth and adoption, while other, technically more elaborate systems, such as hypertext systems from the 1980s, lacked the capabilities for mass contribution and gradual degradation. The lesson to be learned is that a state-of-the-art system that produces higher quality output (e.g., no dangling links) may be less useful to its users than a less consistent system, which achieves a separation of concern, letting users do what they are most concerned about, that is, easily create and access documents. Furthermore, a distributed system without the need for a central coordinator is inherently robust. While there are many potential problems that may affect individual nodes in the World Wide Web, the only problem leading to brittleness in the World Wide Web as a whole is the hierarchical control of the IP addresses and the Domain Name System of the Internet.

2.2.2 Requirements for an Architecture for the Web of Data

Over time, a variety of architectures have been proposed for publishing data on the Web. Many requirements were derived from the design decisions that worked well for the World Wide Web and led to its phenomenal growth, but which had yet to be realized for data and knowledge systems.

In fact, traditional knowledge systems have already exhibited some of the functional requirements sought from the Semantic Web. However, traditional knowledge systems exhibited a lack of flexibility, robustness, and scalability. To quite some extent the problem had been a lack of maturity in the face of algorithmic methods with high computational complexity. For instance, description logics systems, which are now the backbone of Web Ontologies, were severely limited in scale, typically capable of handling not more than a few hundred concepts in the mid-1990s (cf. [40]). Such problems have been assuaged using much increased computational power and better understood and optimized algorithms. However, several bottlenecks remain, which are akin to the problems that the architecture solved for the domain of hypertext documents.

Remaining barriers for managing data and semantics revolve around issues concerning the large number of data sources with varying (1) underlying technologies, (2) geographically dispersed locations, (3) authorities, (4) service quality, and (5) adoption rate. These are exactly the dimensions that had and have to be considered for the design of the World Wide Web.

Thus, in analogy to the World Wide Web, the Semantic Web requires a computing mega system with the following five characteristics:

1. **Explicit, Simple Data Representation:** A common data representation should hide the underlying technologies and only capture the gist of the underlying data representations. Here, the analogy may be drawn with HTML documents that have served as simple, yet effective representations of what constitutes a document.
2. **Distributed System:** The system should be fully distributed, comprising of data sources without a centralized instance that controls who owns what type of information. Distributed ownership and control, if done properly, facilitates adoption and scalability, which is in analogy to websites and Web pages that are under full control of their producers.
3. **Cross-referencing:** In order to benefit from the network beyond the mere sum of its parts, the data must be cross-linked, allowing for reuse of existing data and existing data definitions from different authorities, analogous to hyperlinks allowing for the reuse of text in the hypertext space.
4. **Loose Coupling with Common Language Layers:** In a mega system, the components have to be only loosely coupled. The loose coupling is achieved by communicating in standardized languages. The standardized languages must come with great flexibility such that they may be customized for specific systems, but the overall communication must not be jeopardized by such specialization. The requirement should be seen in analogy to the coupling between different Web clients and servers, where dependency is reduced to understanding HTTP as transport protocol and producing and interpreting HTML content.
5. **Ease of Publishing and Consumption:** The mega system should allow for easy publishing and consumption of simple data and for comprehensive publishing and consumption of complex data. The requirement is in analogy to the Web page description language HTML that provides a simple means of conveying textual information, but that can be viewed, managed, and composed using elaborate browsers and powerful content management systems.

Given these requirements, two points of view for a Semantic Web architecture emerge. One viewpoint is focused on the Semantic Web languages and protocols and is mentioned several times in the above list. Another viewpoint concentrates on the functionalities to be contributed by Semantic Web components.

Requirements for a Semantic Web Language Architecture. At high level of abstraction, Semantic Web languages must address the listed requirements. Below, mandatory objectives are presented, accompanied by examples and, in parenthesis, the requirement they refer to.

First, a data model must be able to represent entities, such as the group called “ABBA,” the person called “Benny Andersson,” their relationship, and the concrete data items, such as the string “Benny Andersson” and the birth date of Benny Andersson (1).

Second, such a data model must be serializable in a standardized manner such that data become easily exchangeable between different computing nodes (1, 2, 4). For instance, without a common data serialization, data from MusicBrainz, BBC, and

Wikipedia or DBpedia cannot be easily joined. A merge of such datasets may lead into forming interesting connections between playlists, music groups, artists, and their origin that span across datasets. While combining data is possible in conventional systems, such as relational databases, the emphasis on the Semantic Web is on the ease of joining such separate pieces of information.

Third, individual entities must be referable in such a data model across borders of ownership or computing systems, thus allowing also for the cross-linking of data (1, 2, 3, 4). Without such cross-linking, “Benny Andersson” from ABBA might be hard to distinguish from the several Benny Anderssons now found on MySpace and Facebook, who also might or might not be musicians.

Fourth, the data model should have an expressive, machine-understandable data description language. In a global data space, having an expressive data description language is of major concern because users and developers can no longer manually inspect and make use of data descriptions due to the sheer size and heterogeneity of data on the Web (1, 5). Furthermore, such a data description language also allows for a refinement of the basic data model, providing levels of specialization needed in the application domains (4). For instance, the richness of BBC program descriptions is hard to understand given long chains of data leading from radio stations, over shows, versions of shows, to the songs which are connected to artists.

Fifth, such a data model requires a query and manipulation language allowing for selections of data or aggregations of data, such as the number of Swedish composers being broadcasted on specific programs (5).

Sixth, reasoning is desirable to facilitate querying, as it provides shortcuts for complex data situations, for example, turning the chain of relationships between a program and a song into a direct relationship using inference rules (5).

Seventh, the transport of data and the transport of queries and results need commonly agreed-upon protocols. While many protocol details are still under discussion, the usage of HTTP is agreed and even refined, for example, for the transport of queries.

Eighth, such a transport requirement may also include encrypted data requests and data transport. Security of data transmission is typically addressed by encrypting the data transmission channel. On the Web, secure data transfer is achieved by HTTPS, a secure version of HTTP using SSL/TLS, established Internet protocols for encrypted data transmission. Beyond the increased security of transport, further security functionality is required, for example, for signing data items. Such features call for a completely distributed authentication system to establish the authenticity of a user request and control access to resources.

Additional Requirements for a Semantic Web Components Architecture. All requirements for Semantic Web languages imply corresponding requirements on functionality to be delivered by software components. However, some software components are not standardized (and should not be), but should be customizable to every individual user needs – up to a point where the community may recognize new chores to be carried out in the stack of software components that should be moved out into a joint language or joint computational model or structure.

Core requirements that are not (yet) included in the language architecture comprise the following.

First, versatile means for user interaction. A Web of Data is not merely a Web of Documents and understanding cross-linked data is different to reading a document. Hence, early means of tabular interfaces constitute a stepping stone, which however is rather a crutch than a full-fledged solution for making the content of the Web accessible to users. Broad accessibility requires viewing, searching, browsing, and querying for data, while at the same time abstracting from the intricacies underlying the distributed origin of the content. The users need the ability to assemble information from a multitude of sources without a priori knowledge about the domain or structure of data. Such on-the-fly integration of multiple data sources enables new interesting scenarios for searching and interacting with the data, but may require software that is oblivious to the underlying structure.

Likewise, interaction should facilitate data production and publishing. Crucial to the success of the Web of Data is that metadata creation and migration of data are made convenient, no matter whether the data originate from the content management systems, relational databases, or competing metadata representations, such as microformats.

Second, the issue of provenance and trust is even more important in a Web of Data than in a Web of Documents. While documents still convey provenance and trust via indications such as authorship and website ownership, corresponding notions for data are watered down once data are processed and aggregated with other data. Hence, notions of origin, reliability, and trustworthiness need to be reconsidered for making them applicable to individual data items and for aggregated datasets. Furthermore, such notions are connected to faithful authentication working at Semantic Web scale.

Third, in exploring the data and weighing their provenance and trustworthiness, one must be able to align unconnected sets of data. Beyond using identifiers, such as URI/IRIs, interlinking implies the capability to suggest alignments between identifiers or concepts from different sets of data. Only with such an alignment, the full picture of a Web of Data may emerge.

2.3 First Principles for a Semantic Web Architecture

A software architecture describes the structure of a software system and serves the purpose of dividing the software into components that may be developed and maintained – and may be used – independently of each other. Thus, a software architecture gives developers, maintainers, and users the possibility to care about their part of the system while being able to communicate system properties with other people.

Unlike traditional information systems design, both the Web and the Web of Data do not emphasize the specific application, but rather generic needs that can be realized in many different components, which only agree on a core set of standards. Thus, the development of the architecture of the Semantic Web first focused on protocols and languages, whereby HTTP (Hypertext Transfer Protocol) was mostly accepted as given for the Semantic Web in

order to be compatible with the Web, but where the language layer was considered pivotal to the flexibility of creating and exploiting a Web of Data.

Therefore, the first architectural viewpoint presented is the famous Semantic Web layer cake which is actually a Semantic Web protocols and languages layer cake. Most of the parts in the layer cake are related to data publishing and exchange.

The second architectural viewpoint considered is a functional view; thus, a description of software components that provide certain functionality, such as supporting language standards or user interaction is given, followed by a discussion of alternative architectural choices for Semantic Web applications.

2.3.1 Protocols and Languages

The Semantic Web is rooted in a set of language specifications which represent a common infrastructure upon which applications can be built. [Figure 2.1](#) illustrates the language standards underpinning the Semantic Web. Unlike most previous variations of the Tim Berners-Lee's Layer Cake (<http://www.w3.org/2007/03/layerCake.png>), the focus here is on the established languages that have been developed in different W3C standardization committees (cf. [Fig. 2.1](#)) and for which stable recommendations are available.

The remainder of the section first briefly introduces each language component, starting at the bottom and progressing to the top; more detailed discussions follow later on.

Given the decentralized nature of the Semantic Web, data publishers require a way to unambiguously refer to resources. Resources on the Internet are identified with Uniform Resource Identifiers (URIs) [5]. URIs on both the Web and the Semantic Web typically use identifiers based on HTTP, which allows for piggybacking on the Domain Name System (DNS) to ensure the global uniqueness of domain names and hence URIs. In the example, the URI <http://www.bbc.co.uk/music/artists/2f031686-3f01-4f33-a4fc-fb3944532efa#artist> denotes Benny Andersson of ABBA. Internationalized Resource Identifiers (IRI) [28] complement URIs and allow for use of characters from a large range of writing systems in identifiers.

Implicit in the use of URIs is a mechanism for retrieving content; assuming an HTTP URI denoting ABBA, a user has the ability to dereference the URI, that is, perform

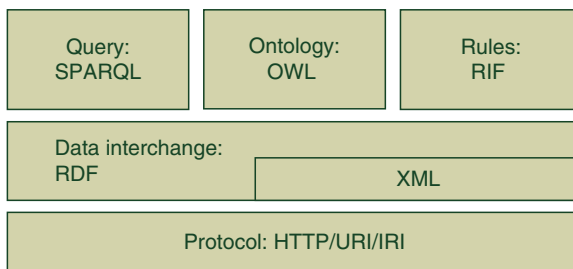


Fig. 2.1

Semantic Web protocol and language standards

a lookup using HTTP. Given there is content hosted at the URI, the user retrieves content which is associated with the URI.

The ability to unambiguously point to resources and dereference them is a first step. Next, required is a language to exchange description of resources. The Extensible Markup Language (XML) is used for encoding documents and provides means for specifying and serializing structured documents which can be parsed across operating systems.


Building on a referencing and a document exchange mechanism, means to encode descriptions about resources are required. Given that the data on the Semantic Web are highly distributed, the description of resources should be encoded in a way that facilitates integration from a large number of sources. A graph-structured data format [61] achieves the easy integration of data from multiple sources. The W3C standard for encoding such data is the Resource Description Framework (RDF). RDF graphs can be serialized in multiple ways; one of the most commonly used is the XML serialization.

Having integrated data, mechanisms for querying the integrated graphs are necessary. SPARQL (a recursive acronym for SPARQL Protocol and RDF Query Language) is a declarative query language (similar to SQL) which allows for specifying queries against data in RDF.

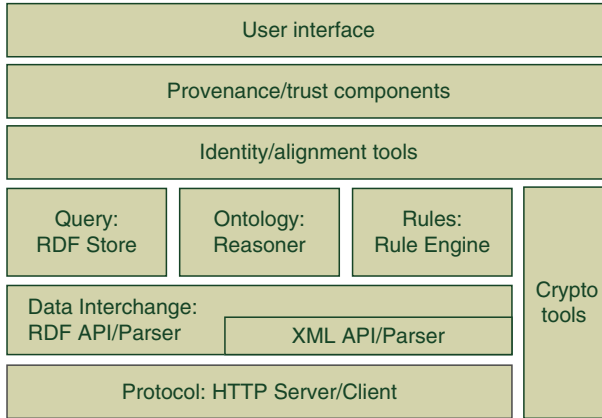
Encoding data as graph covers only parts of the meaning of the data. Often, constructs to model class or property hierarchies provide machines and subsequently humans a more comprehensive understanding of data. To more comprehensively model a domain of interest, so-called ontology languages can be employed. RDF Schema (RDFS) is a language which can be used to express, for example, class and property hierarchies as well as domain and range of properties. Since data originate from multiple sources and are thus highly heterogeneous, means to reconcile data from different sources and to check consistency of combined data are required. The Web Ontology Language (OWL) allows for specifying the equality of resources or cardinality constraints of properties, for example. Ontology languages allow for automated inferences, that is, drawing conclusions based on existing facts.

An alternative way to specifying logical inferences is via rules. Often, users require means to express logical rules which transform data or enrich data with additional specifications. The Rules Interchange Format (RIF) allows for encoding and exchanging such logical rules.

2.3.2 Software Components

One way of viewing the Semantic Web architecture is via the standards and languages used. Another view is via software applications and components that implement functionality based on the standards and languages.  *Figure 2.2* represents the components typically used as Semantic Web infrastructure when implementing functionality in applications. For a survey of Semantic Web applications described in the literature and the components they implement, see [41].

The basic infrastructure components enable the publishing RDF and the dereferencing of content. URIs serve as unique identifiers; however, per convention, URI can be also



■ Fig. 2.2

Semantic Web components architecture

used to retrieve content pertaining to the resources they identify. Web servers provide the infrastructure for serving content via HTTP, and HTTP clients provide lookup functionality. The Semantic Web inherits the basic infrastructure for referencing and lookup from the Web.

Content returned as the result of HTTP lookups can be processed using XML parsers and APIs (in case of non-RDF content) or RDF parsers and APIs. While W3C developed a specification for manipulating XML documents (the Document Object Model), no such standardized specification exists for processing RDF content. However, there are several open-source implementations of APIs for RDF, which are covered in [Sect. 1.4.2](#).

Having gathered RDF data, typical applications use RDF repositories for storing and indexing RDF data and processing SPARQL queries. Ontology languages, such as RDFS and OWL, add more expressivity to RDF data, and so-called reasoners are able to process ontologies and draw specified inferences. Similarly, rule engines allow for processing rules exchanged in RIF.

A crucial point for transmitting sensitive (personal) data is to ensure that data transmissions cannot be intercepted, read, or altered. Crypto-tools cover encryption and authentication technologies to ensure the secure exchange of data. Crypto-modules, such as SSL processors, verify digital certificates and provide cryptographic privacy and authentication.

Given that content aggregated from a large number of sources often uses multiple identifiers to denote the same real-world object, an integration and alignment layer provides for consolidation and the tighter integration of data. The provenance and trust layer analyzes the data in conjunction with additional information to provide the user with a notion of trust associated to individual data items.

Finally, the user interface enables users to interact with Semantic Web data. From a functionality viewpoint, some user interfaces are generic and operate on the graph structure of the data, while others are tailored to a certain domain and ontology.

2.3.3 System Architecture Styles

In general, one can distinguish between two basic architectural styles for applications that consume Semantic Web data [36]: (1) gather and preprocess data a priori, similar to Web search engines [14] and data warehouses; or (2) integrate data on demand at query time, similar to database approaches [54].

In the preprocessing approach, Web crawlers or scripts collect large amounts of data and indexers prepare the entire corpus for fast lookups. The architecture of Web crawlers has been adapted to the intricacies of Semantic Web data (e.g., [37]); also, tools exist that allow for converting metadata embedded in a variety of formats, such as Microformats to RDF. In some of the warehousing systems, reasoning modules ensure that all inferences on the gathered data are computed a priori, and then the queries are answered against the materialized version of the dataset. Typical large-scale Semantic Web search engines [20, 22, 24, 43, 58] use the precomputed approach where all data are collected and preprocessed.

The on-demand query model, on the other hand, is used in meta-search engines [72] and distributed query-processing systems, which start with the query (or goal) and iteratively collect the required data from a set of distributed sources. Meta-search engines collect and integrate results from several search engines during query execution time. Distributed query-processing systems [39, 49] use source descriptions to decide which source can answer parts of a given query and delegate (parts of) the initial query to the appropriate source. In a similar vein, reasoning systems that employ so-called backward-chaining (used in, e.g., XSB [67]) start with a goal and iteratively find rules or data that contribute answers. Semantic Web applications that use the on-demand query model are SemaPlorer [68] and DARQ [64].

Deciding on which architecture to select for applications depends on the requirements of the use case. The warehousing model provides fast query-response times due to the large amount of preprocessing involved, but suffers a number of drawbacks. First, the aggregated data are never fresh as the process of collecting and indexing vast amounts of data is time consuming. Second, from the viewpoint of a single requester with a particular query, there is a large amount of unnecessary data gathering, processing, and storage involved since a large portion of the data might not be used for answering that particular query. Furthermore, due to the replicated data storage, the data providers have to give up their sole sovereignty on their data (e.g., they cannot restrict or log access any more since queries are answered against a copy of the data).

On-demand query processing offers several advantages: the system is more dynamic with up-to-date data and new sources can be added easily without time lag for indexing and integrating the data, and the system requires less storage and processing resources at the query-issuing site. The drawback, however, is that such systems cannot give guarantees about query performance since the integration system relies on a large number of possibly unreliable sources, and that potentially the same work has to be done repeatedly.

2.4 Building Blocks

The following section covers the functional software components some of them implementing W3C recommendations which are used for data publishing and consumption.

2.4.1 Referencing, Transport Protocol, and Linked Data Principles

Data access is a vital component of the architecture of the Semantic Web. The successful traditional client–server model has been adapted to the Web at a grand scale where clients can easily connect to and transfer data from a multitude of servers. Documents on servers are interlinked with each other in a decentralized manner, and clients use these links to navigate from server to server. The model has been condensed to the necessary functionality, and loose coupling between one server and another is achieved by (unilaterally) linking from a document on one server to a document on another server. Actually, whether the referenced document is located on the same server or a server on a different continent does not make a fundamental difference in the process of linking.

URI/IRI and HTTP are the core specifications for both the Web and the Semantic Web, and enable decentralized referencing and transport of content. Uniform Resource Identifiers (URI) are used to identify resources on the Web, for example, <http://www.bbc.co.uk/music/artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist> can be used to denote the music group ABBA.

For URIs on the Semantic Web, there exists a distinction between information resources (i.e., those URIs which are Web pages, such as the page describing ABBA) and noninformation resources (i.e., denoting ABBA the music group, or an abstract concept, such as “the integers”) [34]. It is important to not confuse identifiers of documents with identifiers of things, for example, <http://en.wikipedia.org/wiki/ABBA> is the identifier of the document containing a textual description of ABBA while <http://dbpedia.org/resource/ABBA> represents the music group ABBA itself.

Having the ability to reference things via URIs forms the basis for identity on the Web. Sources other than the authoritative source (i.e., the source which is associated to a URI via syntactic or protocol means) can reuse external URIs, which helps to construct a Web of Data. For example, the content about ABBA at the BBC may reuse DBpedia’s URI for ABBA, thus establishing an association between its own notion of ABBA and the notion of ABBA from DBpedia.

A URI starts with a scheme name (e.g., `http`, `ftp`, `tel`, `mailto`) followed by additional information. The example in this chapter only uses URIs with the Hypertext Transfer Protocol (HTTP) [31] scheme. HTTP URIs have the benefit that when dereferenced – that is, when a HTTP request on them is performed – they return some form of content. In the Semantic Web, the content returned is typically in RDF. HTTP allows for mechanisms to specify additional parameters when requesting content.

For example, clients can ask for content in certain formats to be returned from the Web server using the Accept header, which specifies the preference for certain formats.

Data in the music scenario are published by the BBC, DBpedia, and MusicBrainz. All these data publishers use a subset of the protocols and languages shown in [▶ Fig. 2.1](#). Data are freely interconnectable, meaning that groups can publish data and interlink their data with others' without direct coordination. For example, the BBC links their ABBA URI to the DBpedia equivalent without the need for DBpedia to permit the link or configure anything on their side.

Data publishers on the Semantic Web typically use linked data principles [6]:

1. "Use URIs as names for things.
2. Use HTTP URIs so that people can look up those names.
3. When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL).
4. Include links to other URIs, so that they can discover more things."

Data publishing adhering to these simple principles leads to a unified system for data access and interlinkage. Assuming that all the data publishers use these standards, data consumers can use those standards (as clients) to access the data and further process the data. The so-called linked data cloud of Semantic Web data [21] has been growing considerably in the past years and provides a foundation upon which applications leveraging that data can be built. Wrappers (a special case of mediators [80]) allow for accessing legacy data as linked data. For example, D2R [10] provides a mechanism for exposing data stored in relational databases as linked data. Custom-built wrapper software often provides linked data access to Web sources returning JSON or XML.

HTTP is based on the representational state transfer (REST) [30] request/response model, which can be extended to query/answer for data. This model decouples data providers from data consumers, allowing clients to use uniform interfaces for accessing and querying data stored on remote servers. In the RESTful approach, information about resources identified by URIs can be accessed directly using simple HTTP GET method, or if a site implements a SPARQL endpoint, clients can pose powerful SPARQL queries against the published data. The linked data and data access models are presented in more details in the chapter titled [▶ Semantic Annotation and Retrieval: Web of Data](#).

2.4.2 Data Interchange

Given that data on the Semantic Web are created by people across the planet who do not necessarily coordinate when creating their data, the architecture needs to allow for the distributed creation of content while allowing for integration of and interoperation between the created data. Graph-structured data models (e.g., [61]) have been identified as candidates for integration of data from disparate sources. RDF is such a graph-structured data model that uses URIs as identifying mechanism and provides the means for the separate modeling and publishing of data which can be interlinked and integrated

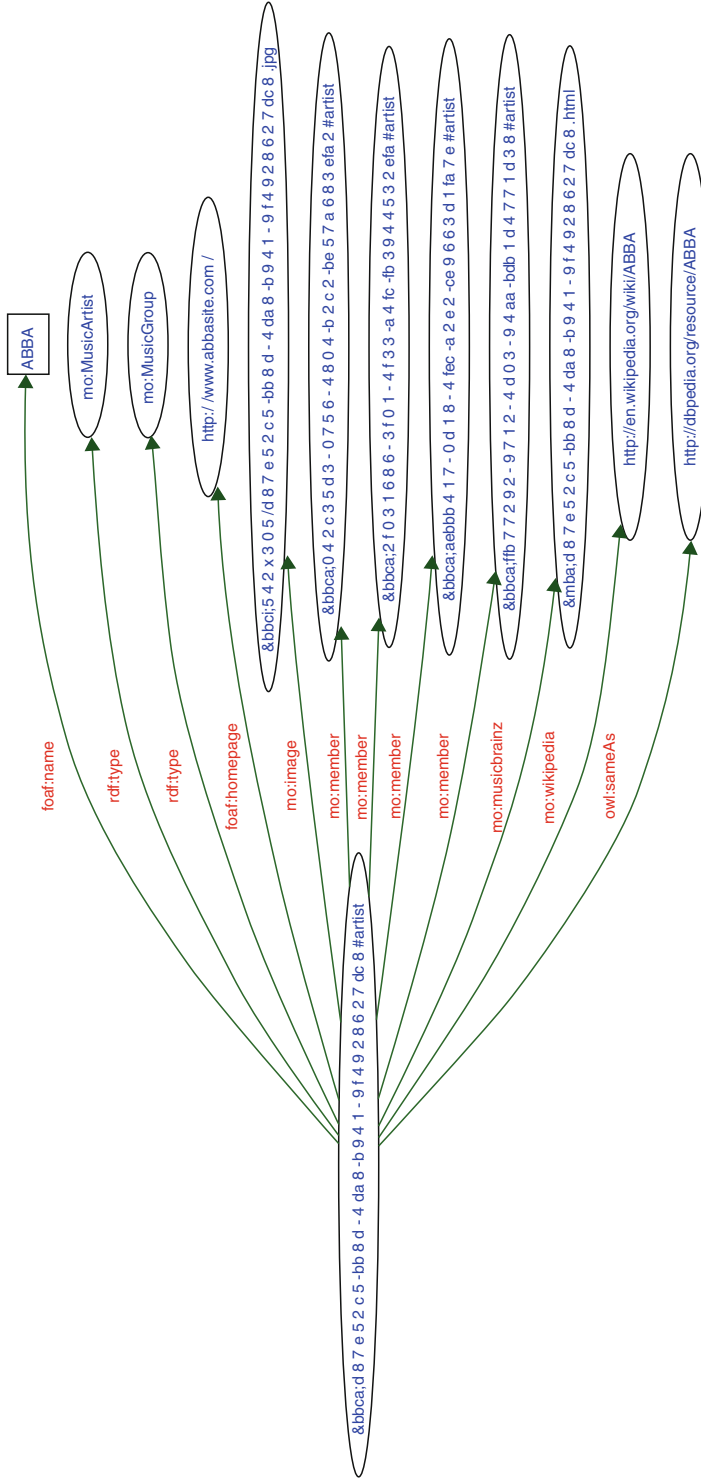
in subsequent steps. RDF is a data format based on directed labeled graphs, which contains data items consisting of (subject, predicate, object) triples. In such triples, subject, predicate, and object can be either URIs or, depending on which position an element is located at, literals (string such as “ABBA” or numbers or dates) or blank nodes – identifiers which are local to a given graph and cannot be referenced from outside. The RDF specification includes a special property `rdf:type` which is used to relate instances (e.g., ABBA) to classes (e.g., the class `MusicGroup`). A graph encoding a description of ABBA is shown in [Fig. 2.3](#).

RDF is an abstract data format which can be written in multiple serializations. One of the serializations is XML, the Extensible Markup Language, which is shown in [Fig. 2.4](#). Other serializations include Turtle, a format used in [Fig. 2.5](#) which encodes RDF in a triple language and allows for shortcuts, such as “;” for repeating subjects or “,” for repeating subject/predicate pairs. A recent development is RDFa which allows for embedding RDF statements directly into (X)HTML documents [1]. To be able to process RDF in application programs, parsers such as ARP from the Jena project [52] or parsers part of the Redland RDF library [4] can be used. These libraries also contain repositories which allow for storing and retrieving RDF. RDF is further described in the chapter titled [Semantic Annotation and Retrieval: RDF](#).

2.4.3 Querying, Updating, and Views

Data access is an important component in the architecture of the Semantic Web. The linked data principles (cf. [Sect. 2.4.1](#)) allow for publishing and accessing simple facts; however, they do not support more complex queries because data published using the linked data paradigm does not have to be accompanied by more sophisticated query mechanisms. Consider the example from MusicBrainz with a query for information about ABBA. Now, consider a query asking for singers from ABBA that were also members of other music bands, as such a situation is not unusual among musicians. Although a software program could navigate from the URI describing ABBA to each of its members, and later to all bands she or he was a member of, iteratively dereferencing individual URIs could be too time consuming for certain use cases. Furthermore, if some of the URIs do not point to real-world Web addresses, it is not even possible to find an answer for such a query.

The SPARQL Query Language for RDF [73] is designed for evaluating queries against RDF datasets and designed to handle complex structure queries, typically over data stored in RDF repositories. The repository that supports SPARQL must implement the querying of the underlying data using a specific syntax and protocol. With RDF repositories, access to the information is no longer realized by dereferencing individual URIs for RDF files, but by posing queries to SPARQL endpoints. SPARQL allows a user to specify arbitrary URIs (even those not accessible on the Web) and a graph pattern that should be matched against the knowledge base together with additional constraints. The example graph pattern for asking about different bands that musicians from ABBA were singing in is presented in [Fig. 2.6](#).



■ Fig. 2.3

RDF graph describing ABBA (partial content of <http://www.bbc.co.uk/music/artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8.rdf> as of December 10, 2009). URIs are abbreviated using RDF/XML syntax from ◀ Fig. 2.4

```

<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
<!ENTITY bbca "http://www.bbc.co.uk/music/artists/">
<!ENTITY bbci "http://www.bbc.co.uk/music/images/artists/">
<!ENTITY mba "http://musicbrainz.org/artist/">
]>

<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:foaf="http://xmlns.com/foaf/0.1#"
  xmlns:mo="http://purl.org/ontology/mo/">

  <mo:MusicArtist rdf:about="&bbca;d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist">
    <rdf:type rdf:resource="http://purl.org/ontology/mo/MusicGroup"/>
    <foaf:name>ABBA</foaf:name>
    <foaf:homepage rdf:resource="http://www.abbasite.com/">
    <mo:image rdf:resource="&bbci;542x305/d87e52c5-bb8d-4da8-b941-9f4928627dc8.jpg"/>
    <mo:member rdf:resource="&bbca;042c35d3-0756-4804-b2c2-be57a683efa2#artist"/>
    <mo:member rdf:resource="&bbca;2f031686-3f01-4f33-a4fc-fb3944532efa#artist"/>
    <mo:member rdf:resource="&bbca;aebbb417-0d18-4fec-a2e2-ce9663d1fa7e#artist"/>
    <mo:member rdf:resource="&bbca;ffb77292-9712-4d03-94aa-bdb1d4771d38#artist"/>
    <mo:musicbrainz rdf:resource="&mba;d87e52c5-bb8d-4da8-b941-9f4928627dc8.html"/>
    <mo:wikipedia rdf:resource="http://en.wikipedia.org/wiki/ABBA"/>
    <owl:sameAs rdf:resource="http://dbpedia.org/resource/ABBA"/>
  </mo:MusicArtist>
</rdf:RDF>

```

■ Fig. 2.4

RDF describing ABBA serialized in RDF/XML

```

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix mo: <http://purl.org/ontology/mo/> .

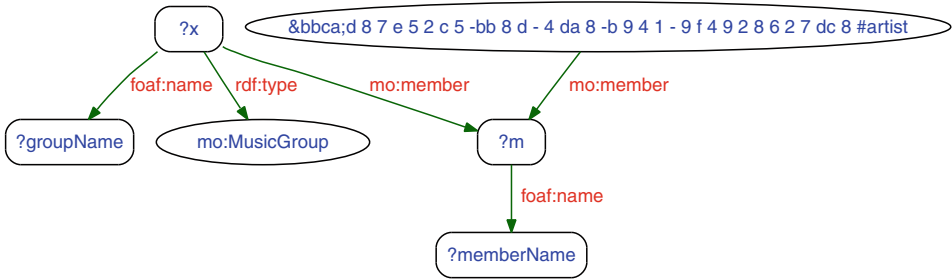
<http://www.bbc.co.uk/music/artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist>
  rdf:type mo:MusicArtist, mo:MusicGroup ;
  foaf:name "ABBA" ;
  foaf:homepage <http://www.abbasite.com/> ;
  mo:image <http://www.bbc.co.uk/music/images/artists/542x305/d87e52c5-bb8d-4da8-b941-9f4928627dc8.jpg> ;
  mo:member <http://www.bbc.co.uk/music/artists/042c35d3-0756-4804-b2c2-be57a683efa2#artist>,
    <http://www.bbc.co.uk/music/artists/2f031686-3f01-4f33-a4fc-fb3944532efa#artist>,
    <http://www.bbc.co.uk/music/artists/aebbb417-0d18-4fec-a2e2-ce9663d1fa7e#artist>,
    <http://www.bbc.co.uk/music/artists/ffb77292-9712-4d03-94aa-bdb1d4771d38#artist> ;
  mo:musicbrainz <http://musicbrainz.org/artist/d87e52c5-bb8d-4da8-b941-9f4928627dc8.html> ;
  mo:wikipedia <http://en.wikipedia.org/wiki/ABBA> ;
  owl:sameAs <http://dbpedia.org/resource/ABBA> .

```

■ Fig. 2.5

RDF describing ABBA serialized in Turtle

The query can be written in SPARQL, as presented in [Fig. 2.7](#). A SPARQL query consists of sections that define different aspects of the query. PREFIX is used to abbreviate URIs, mostly for clarity and to improve readability of the graph pattern. In the SELECT section, users can specify the exact information they are interested in. Alternatively, users can request triples as result to a query using the CONSTRUCT clause. There is no longer



■ Fig. 2.6

Graphical representation of the WHERE clause of the query for music groups that members of ABBA sing in

```

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX mo: <http://purl.org/ontology/mo/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?memberName ?groupName
WHERE { <http://www.bbc.co.uk/music/artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist> mo:member ?m .
       ?x mo:member ?m .
       ?x rdf:type mo:MusicGroup .
       ?m foaf:name ?memberName .
       ?x foaf:name ?groupName }
FILTER (?groupName <> "ABBA")
  
```

■ Fig. 2.7

SPARQL query for music groups that members of ABBA sing in

a limitation to one resource and its description as with linked data lookups on URIs. Please note that the example query shall return only the names of musicians and bands, not the URIs describing them.

The core of a SPARQL query is contained in the WHERE clause. Here, users define the exact graph pattern that has to be matched against Semantic Web data. A basic graph pattern consists of individual (subject, predicate, object) patterns which are joined by variables, forming a template that will be filled during the matching process. In the example, joining resources include the unknown band and members of ABBA. The WHERE clause is used for matching the structure of the graph. Optionally, the WHERE clause is followed by a FILTER expression that narrows the returned results only to the structures that fulfill specific criteria. The example query filters the names of music bands other than ABBA.

SPARQL can be implemented over several kinds of graph repositories. Popular repositories include Sesame [16], Jena [52], Virtuoso [77], BigData [9], OWLIM [60], or RDF-3X [55]. Some of the repositories do not only provide storage and query access for RDF graphs, but also support querying with use of inference and rules. SPARQL lacks a means for expressing inference. It is the role of the underlying repository to support a corresponding inference model.


Since data are often stored in relational databases, wrappers are used to provide SPARQL access to data stored in relational databases or accessible via an API. Examples

include D2R [10] and Triplify [3]. Using such bridges, legacy relational datasets can be exposed and queried as semantic graphs. Such wrappers are of high importance, as they enable smooth transition from the relational model to graph-based processing.

With the recently proposed version of SPARQL 1.1 [47], new features are introduced in the language, which include aggregate functions (such as minimum, maximum, sum), sub-queries (nesting of SPARQL queries), negation and function expressions (e.g., providing a result computed by multiplication of values). Further information about query-specific extensions accompanied by examples can be found at <http://www.w3.org/TR/sparql11-query>.

In addition to the query language, SPARQL defines the access protocol and interoperable data formats. Datasets can be exposed via a SPARQL endpoint to the outside world, accessible via HTTP requests. The access protocols enable remote access to the data and free the data from closed data silos. In the example, MusicBrainz does not only publish the information as linked data, but also provides a query endpoint accessible via HTTP to post SPARQL queries (<http://dbtune.org/musicbrainz/snorql>), which facilitates the use of remote repositories and releases the user from the burden of downloading the dataset locally to be able to pose queries.

Currently, the SPARQL standard provides solution for querying a single knowledge base. It is possible to remotely query MusicBrainz for information about ABBA and later send a query to DBpedia for the biography of each of the band members. Yet to do this, two separate queries have to be asked and the user is in charge of merging the results. New language constructs are proposed in SPARQL to handle the querying of multiple (remote) repositories, but the federated query feature is not defined yet in the standard.

Some solutions to federated queries already have been proposed to address the need of querying multiple repositories. The foundations for querying distributed RDF repositories were stated by Stuckenschmidt et al. [76]. There, the authors proposed a mediation architecture, index structures, and algorithms for executing distributed path queries. The approach was further refined and extended in Networked Graphs [69]. Networked Graphs do not only allow for the querying of remote repositories in a unified manner and building dynamic views of the remote graphs, but also for joining them together and using recursive views (defined as CONSTRUCT queries), and for applying rules. The requirement is that each of the graphs in the query has to be accessible via a SPARQL endpoint. The SPARQL query that uses the Networked Graphs framework for the extended Wikipedia pages of artists singing in ABBA is presented in  Fig. 2.8.

The query accesses two graphs and presents joined results to the user. CONSTRUCT queries produce new RDF graphs that either can be presented to the user or fed as data source to the next query. The implementation of Networked Graphs takes care of distributing proper parts of a query to specific remote SPARQL endpoints (depending on a configuration) and joining the final result. From the user perspective, it is executed as a single query extracting data from two named RDF graphs, hiding the complexity of accessing remote repositories and creating a combined result.

The result of such CONSTRUCT query can be treated as a dynamic, customizable view of the underlying data and allows for presenting data from the connected repositories

```

PREFIX mo: <http://purl.org/ontology/mo/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
CONSTRUCT { ?member mo:wikipedia ?biography . ?member foaf:name ?name }
FROM NAMED :Musicbrainz FROM NAMED :DBpedia
WHERE {
  GRAPH :Musicbrainz {
    <http://www.bbc.co.uk/music/artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist> mo:member ?member .
    ?member foaf:name ?name }
  GRAPH :DBpedia {
    ?member mo:wikipedia ?biography }
}

```

Fig. 2.8

SPARQL query using Named Graphs for Wikipedia pages of ABBA members

in a specified, and even restricted form. With mechanisms such as Named Graphs it is possible to hide the heterogeneity of data sources and schemas, restrict access to specific resources, or redefine types of relationships between entities. The construct can serve the same purpose as SQL views in relational databases. As the view mechanism produces RDF graphs as result, it can be later used as a single data source for constructing another view.

In addition to accessing data, the newly proposed version of SPARQL Update [71] introduces methods for data and graph manipulation in the form of inserts and deletes. This language accompanies SPARQL, using its syntax and protocol to express updates to RDF repository. Until now, all updates to data in the repository had to be performed using storage-specific tools. With the SPARQL Update standardization, changes to the semantic data in repositories no longer require use of third-party applications. Statements can be added or removed from the repository using the SPARQL language. The chapter titled [Querying the Semantic Web: SPARQL](#) contains a more detailed description of the SPARQL query language and protocol.

2.4.4 Ontology and Reasoning

Semantics can emerge in two ways: the first is a social one where meaning arises via a common understanding in a group of people using shared identifiers (covered in [Sect. 2.4.7](#)). The second way for encoding meaning is by the use of logical constructs. The standards enabling reasoning on the Semantic Web are RDF Schema [15], the ontology language OWL (Web Ontology Language) [79], and RIF (Rule Interchange Format) [12], covered in the next section.

To illustrate the use of reasoning, that is, drawing conclusions from existing facts in the Semantic Web, a selection of ontology constructs are explained based on our example, namely, `rdfs:subClassOf`, `owl:sameAs`, and property chains.

The `rdfs:subClassOf` construct can be used to model class hierarchies. Consider, for example, the Music Ontology which specifies two classes, `mo:MusicGroup` and `mo:MusicArtist`, and contains an axiom specifying that `mo:MusicGroup` is a subclass of

`mo:MusicArtist` via the `rdfs:subClassOf` property. Such a construct allows a reasoner to deduce that instances of `mo:MusicGroup` are also of type `mo:MusicArtist`. In the example, given the axiom and a fact stating that ABBA is a `mo:MusicGroup`, a reasoner can draw the conclusion that ABBA is also of type `mo:MusicArtist` and applications that query for all `mo:MusicArtists` also get ABBA as a query result even if the instance does not explicitly contain that type of assertion.

Another construct is that of `owl:sameAs` which can be used to specify that two resources are identical. For example, one could state that <http://www.bbc.co.uk/music/artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist> is the same as <http://dbpedia.org/resource/ABBA>, which allows a reasoner to consolidate information about ABBA from multiple sources. Using `owl:sameAs` instead of reusing the same URI across sources allows for sources being connected after they coined the URIs for the respective thing. In addition, by coining a separate URIs at a source, one may provide information about the resource via HTTP lookups on the URI.

Property chains, which have been added to the recent OWL 2 specification, are useful to collapse property paths. To find out which show played which songs, on the original BBC data one must have access to the timeline of the show, which requires traversing a path of four properties. Using an axiom as shown in [Fig. 2.9](#), a reasoner can collapse that lengthy path into a single predicate, facilitating access to data via the specified shortcut.

OWL enables also more complex modeling constructs, for example, cardinality restrictions (a predicate can only have a certain number of objects) or disjoint classes (the set of instances of two classes are disjoint). Specialized software systems, so-called reasoners, are able to take OWL knowledge bases and (1) check for the consistency of the knowledge base and (2) infer new statements based on existing ones. More details on ontology reasoning can be found in the chapter titled [KR and Reasoning on the Semantic Web: OWL](#).

2.4.5 Rules and Rule Engines

Another mechanism for drawing conclusions from existing knowledge are logical rules (e.g., known from Prolog). Rules consist of two parts, antecedent and consequent: if the statement in the antecedent is true, the statement in the consequent follows. RIF is the

```
<rdf:Description rdf:about="#hasTimeline">
  <owl:propertyChainAxiom rdf:parseType="Collection">
    <owl:ObjectProperty rdf:about="&dc;relation"/>
    <owl:ObjectProperty rdf:about="&po;version"/>
    <owl:ObjectProperty rdf:about="&po;time"/>
    <owl:ObjectProperty rdf:about="&timeline;timeline"/>
  </owl:propertyChainAxiom>
</rdf:Description>
```

■ Fig. 2.9

OWL2 Axiom encoding a property chain

```

if { ?x foaf:firstName ?first;
    foaf:surname ?last }
then
  { ?x foaf:family_name ?last;
    foaf:givenname ?first;
    foaf:name func:string-join(?first " " ?last)
  }

if { ?x foaf:name ?name } and
    pred:contains(?name, " ")
then
  { ?x foaf:firstName func:string-before(?name, " ");
    foaf:surname func:string-after(?name, " ")
  }

```

▣ Fig. 2.10

Two RIF rules for mapping FOAF predicates

W3C recommendation for exchanging rule sets between rule engines. The common subset of a large number of rules systems is standardized in RIF Core [12] which forms the basis for more expressive languages, notably Prolog-style and Production Rule languages. [▶ Figure 2.10](#) shows two mapping rules relating predicates in the FOAF vocabulary [38]. Rules are presented in more detail in chapter titled [▶ KR and Reasoning on the Semantic Web: RIF](#).

2.4.6 Security and Encryption

In open systems such as the Internet, where data are transferred across unsecured links involving infrastructure maintained by a large number of organizations, mechanisms for secure exchange of data have to be established. In addition, technology has to be in place to ensure the authenticity of documents. Further, to establish the identity of users, authentication mechanisms are required. These issues are addressed in the architecture of the (Semantic) Web in the crypto layer.

To make sure data are not altered during transmission, HTTPS, a secure version of HTTP, has been developed [66] using a different server port compared to HTTP and an encryption protocol to counteract man-in-the-middle attacks and eavesdropping of connections on the level of the transport layer.

Digital signing of RDF graphs ensures the authenticity of content – that the content at hand has been created by a known source and that the content has not been altered. Signing RDF documents consists of a normalization step which returns a canonical representation of RDF triples which then are signed using standard digital signature methods [17].

For establishing the identity of a user when logging into a website or service, two similar mechanisms are currently in use. With OpenID [57], users are redirected from a service provider to an identity provider which authenticates the user and returns the credentials of the user to the service provider. In contrast to OpenID, FOAF+SSL [75] is a distributed mechanism based on digital certificates and RDF which allows for

authentication using information from published and signed FOAF profiles. In this model, the FOAF friendship network backed with the cryptographic keys and signatures provides information for the successful authentication of users identified by their FOAF URI.

2.4.7 Identity Management and Alignment

Identity is related to the social meaning of semantics and concerns the question of what identifiers mean and how to ensure the authenticity of an identifier (i.e., how to know an identifier is “the right one” to stand for a real-world entity). The issue of identity is substantial in the context of the Semantic Web, given that the system is decentralized and people may mint own identifiers for resources, which leads to a plethora of identifiers for the same real-world entity. Reusing identifiers across sources allows for discovery and navigation in such a decentralized environment. Using instance URIs or class URIs coined by other sources helps to weave a Web where distributed discovery is possible. For example, the BBC linking to <http://dbpedia.org/resource/ABBA> establishes an association with BBC’s description of ABBA and the one at DBpedia.

Indications for identity in the sense of object equality can be explicitly stated via reusing identifiers across sources, either directly or via connecting an own resource with an already existing one (e.g., via `owl:sameAs` properties on the instance level or `rdfs:subClassOf` on the class level).

On the Semantic Web, anybody can use RDF descriptions to attach certain pieces of information to newly minted or existing URIs. To be able to consolidate information from multiple sources, the identity of URIs has to be established. In the simplest case, when two sources attach RDF descriptions to the same URI, a syntactic check can be used to merge the data. At data creation time, the data publishers have to be aware of the existence of a URI (and what entity it refers to) and therefore decide that they mean the same thing, reuse the URI, and thus subscribe to the meaning of the URI. The data publishers can then attach new descriptions to the URI. For example, to add a new photo of ABBA one could simply publish a triple as shown in [Fig. 2.11](#). Systems such as <http://sameas.org/> or the Okkam Entity Server [13] provide services for determining equivalent identifiers. For a given identifier, if it is known to the system, they provide a list of co-referant URIs. They make it possible to locate the same entity in different sources, although an entity may be referred to by multiple different URIs.

Data publishers can also relate their identifiers to already existing ones, which is in-line with linked data principles. Interlinking data and reusing concepts already modeled in

```
<http://dbpedia.org/resource/ABBA> foaf:depiction
  <http://farm3.static.flickr.com/2137/2203448820_c66459d45a_o_d.jpg> .
```

Fig. 2.11

Annotating an existing URI

existing ontologies facilitates data integration and reasoning. The BBC publishing a triple linking their concept of the instance ABBA to the one at DBpedia via the `owl:sameAs` property is such an example. A case of relating own identifiers to already existing ones on the data description level is the Music Ontology stating that `mo:MusicArtist` is an `rdfs:subClassOf foaf:Agent`. By that, the creators of the Music Ontology relate their concept of `MusicArtist` to the more general and established concept of `foaf:Agent`.

Another way to establish the sameness of identifiers is via reasoning. For example, OWL allows for specifying so-called inverse functional properties which uniquely identify a thing, such as an ISBN for books or passport numbers for people. Reasoners can be used to establish the sameness of two URIs if they share the same value for a property which is defined as inverse functional.

While it is desirable to have associations between available data sources, disparate sources often lack links between them. To allow for the querying and processing data from these sources in an integrated manner, mappings between identifiers have to be found. The Silk [78] linking framework allows for a declarative specification of how resources relate to each other and thus provides a semiautomatic way of matching identifiers across sources. Other approaches, such as [18], allow for automated instance matching based on defined similarity metrics.

Work on ontology matching has mostly focused on aligning class- and property-level identifiers. For example, [27] presents algorithms to align taxonomies with each other. For a comprehensive review of state-of-the-art methods for ontology matching, see [29]. The Ontology Alignment Evaluation Initiative [56] holds annual events to evaluate systems and publish matching accuracy of participating systems.

The graph-structured nature of RDF allows for the amalgamation of data from disparate sources. If the data sources do not share identifiers between each other, instance and ontology matching can provide the tight integration that more elaborate application scenarios require.

2.4.8 Provenance and Trust

On the Semantic Web, not all data are created equal, and in order to judge the value of the data items and how to use data appropriately, one has to know the origin of data. Data provenance can be traced back in various ways. Along formal chains of information processing, such as queries, one may trace the data lineage (as data provenance is called under these specific circumstances) using a formal machinery [25, 32]. These models can provide explanation to questions, such as which pieces of data were chosen and composed together, where do they come from, who created them, or which inference rules were used to create implicit statements.

Along chains of data processing, where the semantics of the processing are not fully specified, one needs more abstract models to trace data provenance. Such a more abstract model is given by workflow provenance models, such as the Open Provenance Model (OPM) [53]. In these models, stages of data processing can also be black boxes (unlike in

data lineage tracing), and provenance includes also information about processes constraints, interactions, and dependencies. Such meta-information is crucial for understanding what happened in the whole chain of data manipulation, which process produced certain results, or who initiated them and when. An example of a workflow provenance with inference (dotted lines) in the Open Provenance Model is shown in [Fig. 2.12](#).

Based on provenance one may also attribute other properties, such as trust on data. There are multiple approaches to modeling trust and trust propagation. In data lineage, trust is related to information such as data source, authorship, certainty, and other dimensions of meta-knowledge. Queries that combine data of different certainty or from multiple data sources need a specific algebra to calculate the overall trust of a result. Such a comprehensive algebra for querying and reasoning with different dimensions of meta-knowledge is described in [26]. Using the proposed algebra, certainty can be calculated for multiple joined statements and graphs, providing a trust value for larger structures.

In peer-to-peer networks, peers must decide if a node is reliable or malicious. Trust is related to the reputation of the node, and modeling requires both a notion of trust and distrust. Trust to the specific node (reputation) reflects the experiences of all peers in the network interacting with the node and each peer gains a view of the network that is wider than its own experience [45].

In social networks, trust follows the friendship relationship. Users define their level of trust to the direct neighbors. An example of Me trusting Alice and Bob, and Alice and Bob trusting Carol is shown in [Fig. 2.13](#). Trust to an unknown person (Carol) is based on trust defined in own friendship network, and further on trust of friends of their friends [33].

Provenance and trust provides the way for verification of data and information. In an open environment, such as Semantic Web, where everybody can publish any data without limitation, provenance can help to distinguish fake from the genuine data.



Fig. 2.12

Inference in the Open Provenance Model

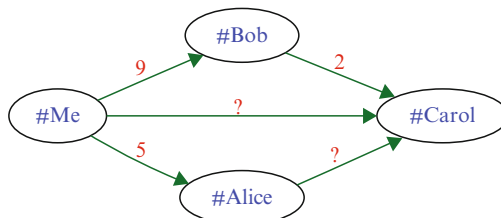



Fig. 2.13

Example of friendship relations in a user network with assigned trust values

2.4.9 User Interaction and Visualization

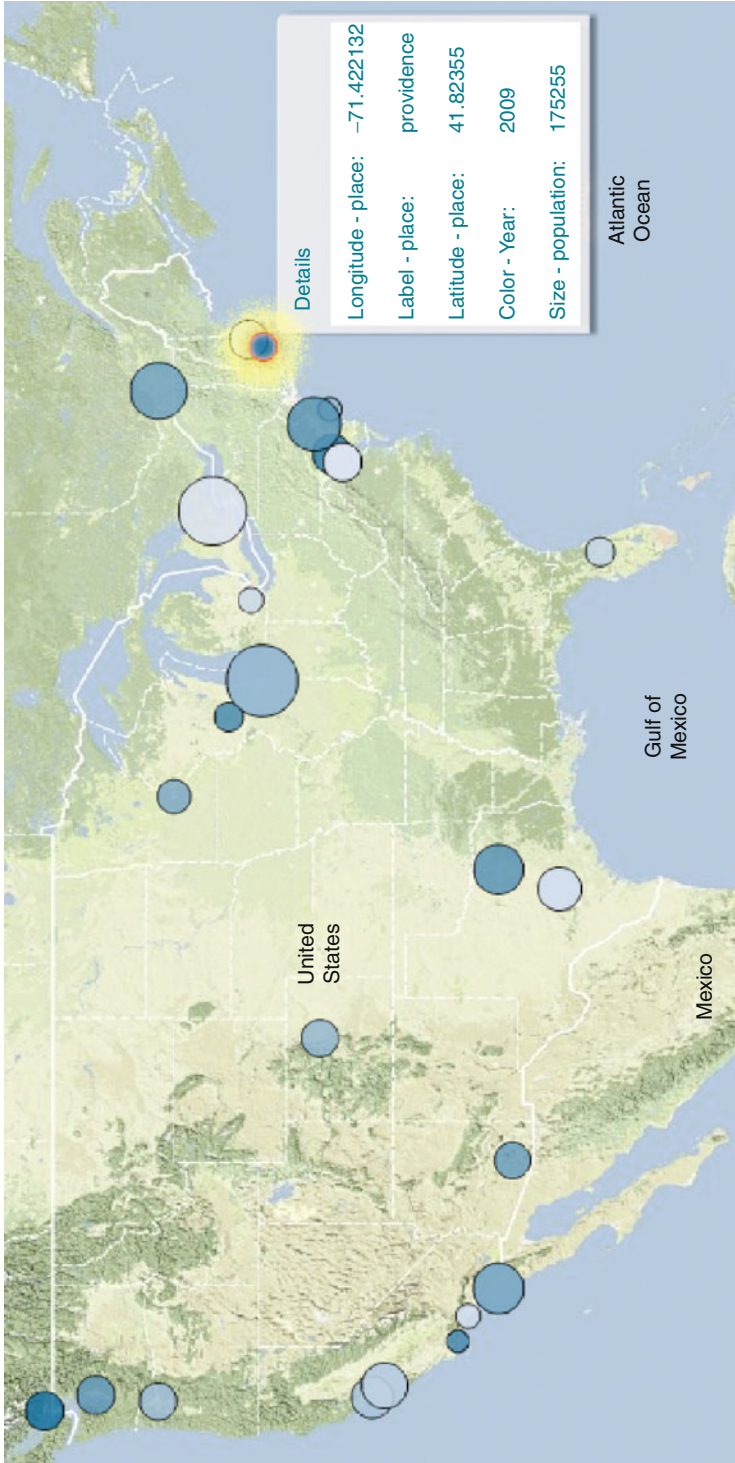
User interfaces over Semantic Web data are still in their infancy, given that sizable amounts of quality RDF data have only been recently made available online. Current challenges for building user interfaces over Semantic Web data are: interacting in new ways, heterogeneous data, and data creation [70]. Dealing with data rather than documents (as in traditional Web search engines) poses several new challenges for interaction design. Data collected from thousands of sources on the Web exhibits far greater variance in interlinkage and vocabularies used than relational or single-source datasets. In addition, a Semantic Web user interface should allow users to change and create new data rather than just allow for viewing and browsing. In the following, each challenge is discussed in detail.

The basic paradigm for interacting with data is query/answer: users construct queries and the system returns answers. Thus, a system enabling users to interact with data has to provide means for posing queries and displaying the results that the system returns. General-purpose user interfaces can offer access to data using keywords, natural language, iterative menu-guided mechanisms, or graphical query languages [46]. Iterative menu-guided systems are used in the majority of current user interfaces over Semantic Web data. In such systems, users can build queries of different expressiveness. While systems such as the Disco Hyperdata Browser [11] allow for traversing the RDF graph one resource at a time, other systems implement more expressive query functionality, such as faceted browsing [35, 48, 59, 74, 81].

Since the answer that these systems return comprises data – typically a subgraph in the case of Semantic Web data – the results can be displayed using appropriate visualizations. Vispedia [19] is a Web-based visualization system operating over data from DBpedia in an interactive, interactive data exploration process, which enables a broad class of nonexpert users to use data extracted from Wikipedia.  *Figure 2.14* shows a map visualization of the North American cities hosting a SIGMOD conference.

User interfaces over Semantic Web data can be organized along a continuum from systems that are coded against a predefined ontology (e.g., SEAL [51]) to generic user interfaces which assume graph-structured data as underlying data model, with minimal knowledge about the kind of data on which they operate (e.g., Tabulator [7]). Since the Semantic Web data may include ontologies which specify the meaning of data, and the data model is universal [23], there is the possibility for creating a single type of client which aims to provide seamless access to arbitrary content. For dealing with content heterogeneity, general-purpose user interfaces have to offer generic interaction and visualization functionality, while being extensible and offering specialized capabilities for data from domains which were not anticipated during design-time of the interface [63].

Systems may use a declarative language to specify how RDF should be rendered, similar to using XML style sheets to render XML documents. Fresnel [62] allows for the specification of so-called lenses which describe how a given piece of RDF should be displayed. Fresnel lenses are used to select and order parts of an RDF graph, and Fresnel formats add content formatting and hooks to CSS styling instructions. RDF graphs often



■ Fig. 2.14

A map visualization of data covering location of SIGMOD host cities in North America and their populations (from [19])

encode complex data which require dedicated visualization widgets for interaction and presentation. An example of the display of complex data is the SIMILE Timeline widget [44] which can be used to visualize temporal information encoded in an RDF graph.

A major task in building domain-independent user interfaces over Web data is to decide on how to order query results. As in Web search, ranking plays an important role here and allows the software to decide on orderings in lieu of a fixed schema which is typically used to decide on ordering data items. These systems typically employ ranking to prioritize and order items for display.

Finally, Semantic Web user interfaces should allow for the creation and editing of content. The Semantic MediaWiki [50] software extends the wiki system powering Wikipedia with a syntax for adding RDF to wiki pages. Tabulator Redux [8] adds editing capabilities to the generic data browser.

The Semantic Web brings a new set of challenges to user interface design. Easy-to-use interfaces are crucial to achieving the scenario outlined in the beginning of the chapter and enables users get answers to questions such as “Which type of music is played in UK radio stations?” and “Which radio station is playing titles by Swedish composers?”

2.5 Related Resources

1. Berners-Lee, T.: Semantic Web. In: Presentation at XML 2000 Conference, Washington, DC. <http://www.w3.org/2000/Talks/1206-xml2k-tbl/> (2000)
This presentation gives an overview of the Semantic Web, its philosophy, planned architecture, and the basic building blocks. It introduces the layers of the Semantic Web (Semantic Web layer cake), highlights the differences between the Web and Semantic the Web, and explains core architecture layers. Additionally it shows directions for the practical deployment of the Semantic Web.
2. Hendler, J.: My take on the Semantic Web layer cake. In: Presentation at Dagstuhl Semantic Web Conference, Germany (2009)
<http://www.cs.rpi.edu/~hendler/presentations/LayercakeDagstuhl-share.pdf>
A funny dinnertime presentation of the Semantic Web layer cake in rhymes from the Dagstuhl conference. Jim Hendler presents the Semantic Web layer cake, how and from where it evolved, and provides some details on different layers – all written as funny, rhymed story. It provides an overview of the core Semantic Web layers placing them in the contexts of previously defined and currently existing Web layers or stacks.
3. Hitzler, P., Krötzsch, M., Rudolph, S.: Foundations of Semantic Web Technologies. Chapman & Hall/CRC Textbooks in Computing (2009)
The authors present a comprehensive description of foundations for the Semantic Web. The major focus is on ontology languages and representation, formal semantics, logic, rules, and reasoning. In addition to detailed chapters on foundation issues, authors present highlights of ontology engineering and example applications. The book provides basics for understanding each of the presented building blocks of the Semantic Web.

4. Staab, S., Studer, R. (eds.): *Handbook on Ontologies*. Springer (2009)
The *Handbook on Ontologies* is a collection of papers by several authors that covers different aspects of ontology engineering. The Handbook begins with the ontology basics like formal languages, logics and reasoning, then presents methodology and different approaches for ontology engineering. Further, it concentrates on the use of a wide range of existing ontologies, from top-level and domain ontologies to task or application specific. Finally, it describes an infrastructure that enables the efficient use of ontologies, reasoning, their storage and retrieval, and presents multiple applications and methods for the use of ontologies in different tasks and applications.
5. Artz, D., Gil, Y.: A survey of trust in computer science and the Semantic Web. *Journal of Web Semantics – Science, Services and Agents on the World Wide Web* (2007)
This survey presents different approaches to modeling trust and reasoning about trust. It identifies four major areas of research in trust: policy-based, reputation-based, general models, and trust in information resources. Each of the identified areas is described in more detail, to give the reader an overview of its most important methods and approaches. The authors provide an extensive list of thematically organized references to publications on trust over the last decade.

2.6 Future Issues

This chapter presented an overview of the architecture of the Semantic Web, starting from the basic principles and requirements, further describing its basic building components with their functionality, ending with approaches to semantic user interfaces. The Semantic Web extends the existing Web and makes online content easier to process by computer programs. Information is represented using explicit statements (facts), with specific semantics that link different pieces of information together. The Semantic Web brings additional meaning to the information on the Web and allows for complex applications operating over collaboratively edited data.

There are multiple layers and services defined within the architecture of the Semantic Web. The lower layers specify transport and serialization principles using XML and RDF. On top of them, languages such as RDFS and OWL allow for the expression of additional semantics and relationships within the data. They introduce schema, distinction between objects and concepts, restrictions, entailment, and reasoning rules. Using these ontology languages, it is possible not only to state simple facts, but also to check the correctness and consistency of the defined ontology or infer additional facts.

In parallel, the Semantic Web model defines data access principles based on the notion of linked data, in addition to a query language, SPARQL, which allows for accessing and querying knowledge bases in a unified manner, abstracting from the underlying complexity or reasoning mechanisms. W3C recommendations represent a stable technical foundation for data publishing. Although their adoption rate is growing, a direct and comprehensive reuse of published data remains an open issue [42]. Other currently open

research questions relate to the use of cryptographic methods for authentication and encryption and the tracking of provenance of data.

Higher layers and services provide extended semantics. Several provenance models are used to provide links to the origin and history of the data, while trust defines the trustworthiness of simple, compound, and derived information. Rules can be used to model additional dependencies and define inference beyond the axioms defined in ontology languages. Such layers extend the standard Semantic Web and make it more attractive for numerous applications.

Finally, information from the Semantic Web has to be exchanged in a secure manner and presented in a human understandable form to the user. Mechanisms such as HTTPS and FOAL+SSL ensure the security of data transportation and an appropriate access mechanism. A variety of applications provide specialized graphical interfaces for searching and navigating the Semantic Web data in a user-friendly way.

Layers and services overviewed in this chapter characterize different elements of the Semantic Web and define a composite system that covers multiple aspects: from data representation via integration and inference to visualization. The components introduced in this chapter are explained in more detail in the following chapters of this book.

2.7 Cross-References

- ▶ KR and Reasoning on the Semantic Web: OWL
- ▶ Querying the Semantic Web: SPARQL
- ▶ Semantic Annotation and Retrieval: RDF
- ▶ Semantic Annotation and Retrieval: Web of Data

References

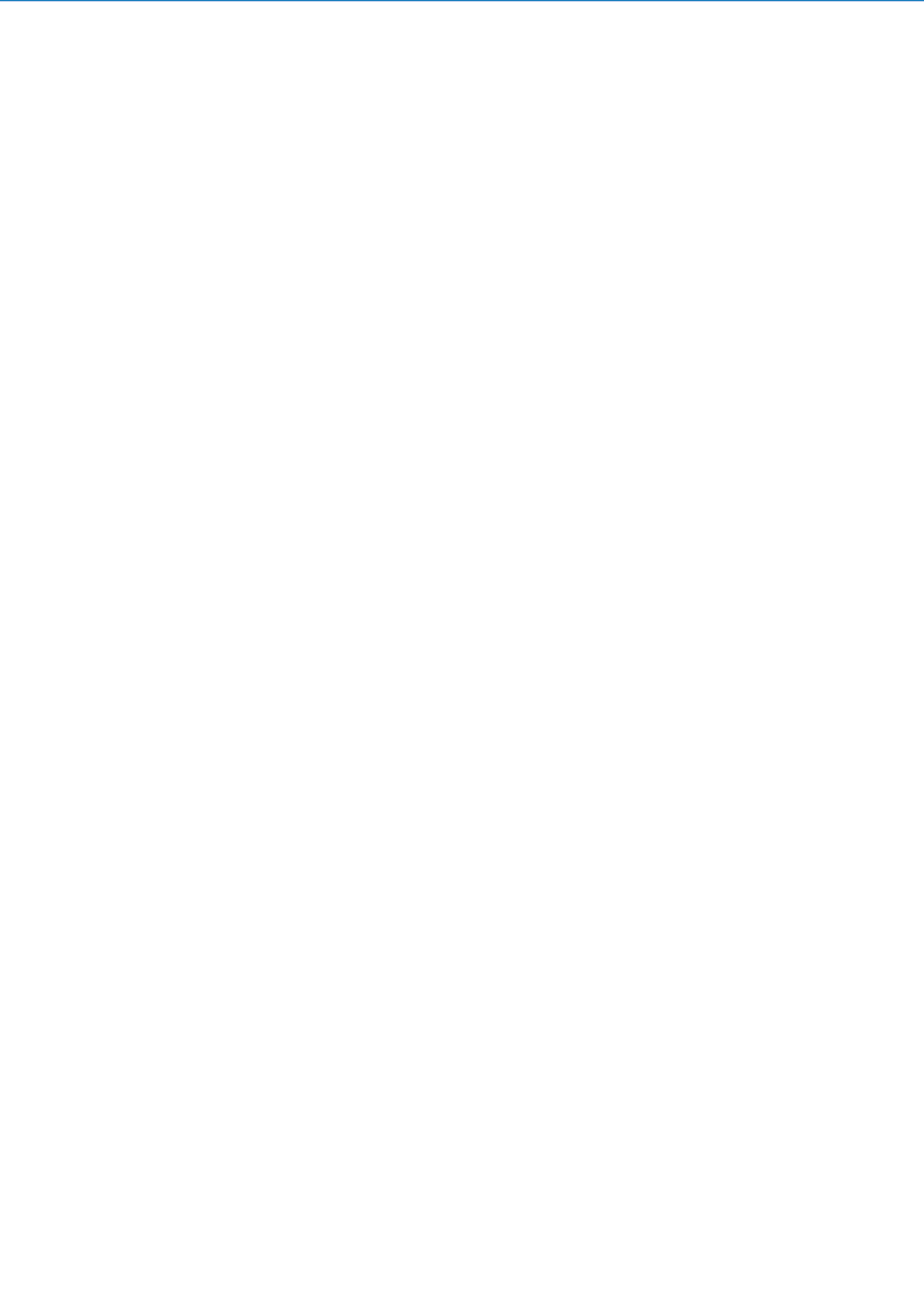
1. Adida, B., Birbeck, M.: RDFa primer. W3C working group note, W3C (Oct 2008)
2. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: a nucleus for a web of open data. In: Proceedings of Sixth International Semantic Web Conference, Second Asian Semantic Web Conference (ISWC + ASWC 2007), Busan. Lecture Notes in Computer Science, Vol. 4825, pp. 722–735. Springer, Berlin (2008)
3. Auer, S., Dietzold, S., Lehmann, J., Hellmann, S., Aumueller, D.: Triplicify: light-weight linked data publication from relational databases. In: Quemada, J., León, G., Maarek, Y.S., Nejdil, W. (eds.) Proceedings of the 18th International Conference on World Wide Web (WWW 2009), Madrid, pp. 621–630. ACM, New York (2009)
4. Beckett, D.: The design and implementation of the redland RDF application framework. *Comput. Netw.* **39**(5), 577–588 (2002)
5. Berners-Lee, T.: Linked data – design issues. <http://www.w3.org/DesignIssues/LinkedData> (2006)
6. Berners-Lee, T.: Universal resource identifiers in WWW: a unifying syntax for the expression of names and addresses of objects on the network as used in the world wide web, RFC 1630. Internet Engineering Task Force. <http://www.ietf.org/rfc/rfc1630.txt> (1994)
7. Berners-Lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J., Lerer, A., Sheets, D.: Tabulator: exploring and analyzing linked data on the semantic web. In: Proceedings of the Third

- International Semantic Web User Interaction Workshop (SWUI 2006), co-located with International Semantic Web Conference (ISWC 2006), Athens, GA (2006)
8. Berners-Lee, T., Hollenbach, J., Lu, K., Presbrey, J., Prud'ommeaux, E., Schraefel, M.: Tabulator Redux: browsing and writing linked data. In: Proceedings of the Linked Data on the Web (LDOW 2008), Beijing, co-located with the 17th International World Wide Web Conference (WWW 2008) (2008)
 9. BigData: Bigdata RDF database. <http://www.bigdata.com/>
 10. Bizer, C., Gau, T.: Disco – hyperdata browser. <http://www4.wiwiw.fu-berlin.de/bizer/ng4j/disco/>
 11. Bizer, C., Cyganiak, R.: D2rq – lessons learned. In: W3C Workshop on RDF Access to Relational Databases. <http://www.w3.org/2007/03/RdfRDB/papers/d2rq-positionpaper/> (2007)
 12. Boley, H., Hallmark, G., Kifer, M., Paschke, A., Polleres, A., Reynolds, D.: RIF core dialect, W3C Recommendation, World Wide Web Consortium (W3C). <http://www.w3.org/TR/rif-core/> (Jun 2010)
 13. Bouquet, P., Stoermer, H., Mancioffi, M., Giacomuzzi, D.: Okkam: towards a solution to the “identity crisis” on the semantic web. In: Semantic Web Applications and Perspectives. Proceedings of the Third Italian Semantic Web Workshop (SWAP 2006), Scuola Normale Superiore, Pisa (2006)
 14. Brewer, E.A.: Combining systems and databases: a search engine retrospective. In: Readings in Database Systems, 4th edn. Morgan Kaufmann, San Francisco (1998)
 15. Brickley, D., Guha, R.: RDF vocabulary description language 1.0: RDF schema, W3C Recommendation, World Wide Web Consortium (W3C). <http://www.w3.org/TR/rdf-schema/> (Feb 2004)
 16. Broekstra, J., Kampman, A., Harmelen, F.V.: Sesame: a generic architecture for storing and querying RDF and RDF schema. In: Proceedings of the First International Semantic Web Conference (ISWC 2002), Sardinia. Lecture Notes in Computer Science, vol. 2342, pp. 54–68. Springer, Berlin (2002)
 17. Carroll, J.J.: Signing RDF graphs. In: Proceedings of the Second International Semantic Web Conference (ISWC 2003), Sanibel Island. Lecture Notes in Computer Science, vol. 2870, pp. 369–384. Springer, Berlin (2003)
 18. Castano, S., Ferrara, A., Montanelli, S., Varese, G.: Matching semantic web resources. In: Proceedings of the 20th International Workshop on Database and Expert Systems Applications (DEXA 2009), Linz, pp. 84–88. IEEE Computer Society, Los Alamitos (2009)
 19. Chan, B., Talbot, J., Wu, L., Sakunkoo, N., Cammarano, M., Hanrahan, P.: Vispedia: on-demand data integration for interactive visualization and exploration. In: Proceedings of the 35th SIGMOD International Conference on Management of Data (SIGMOD 2009), Providence, pp. 1139–1142. ACM, New York (2009)
 20. Cheng, G., Qu, Y.: Searching linked objects with falcons: approach, implementation and evaluation. *Int. J. Semant. Web Inf. Syst.* 5(3), 49–70 (2009)
 21. Cyganiak, R.: About the linking open data dataset cloud. <http://richard.cyganiak.de/2007/10/lod/>
 22. d’Aquin, M., Baldassarre, C., Gridinoc, L., Angeletou, S., Sabou, M., Motta, E.: Characterizing knowledge on the semantic web with Watson. In: Proceedings of Fifth International Workshop on Evaluation of Ontologies and Ontology-bases tools (EON 2007), co-located with the International Semantic Web Conference (ISWC 2007), Busan, pp. 1–10 (2007)
 23. Decker, S., Melnik, S., van Harmelen, F., Fensel, D., Klein, M.C.A., Broekstra, J., Erdmann, M., Horrocks, I.: The semantic web: the roles of XML and RDF. *IEEE Internet Comput.* 4(5), 63–74 (2000)
 24. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V.C., Sachs, J.: Swoogle: a search and metadata engine for the semantic web. In: Proceedings of the 13th ACM International Conference on Information and Knowledge Management (CIKM 2004), Washington, DC (2004)
 25. Dividino, R.Q., Schenk, S., Sizov, S., Staab, S.: Provenance, trust, explanations – and all that other meta knowledge. *Künstl. Intell.* 23(2), 24–30 (2009)
 26. Dividino, R.Q., Sizov, S., Staab, S., Schueler, B.: Querying for provenance, trust, uncertainty and other meta knowledge in RDF. *J. Web Semant.* 7(3), 204–219 (2009)
 27. Doan, A., Madhavan, J., Dhamankar, R., Domingos, P., Halevy, A.: Learning to match

- ontologies on the semantic web. VLDB J. **12**(4), 303–319 (2003)
28. Duerst, M., Suignard, M.: Internationalized Resource Identifiers (IRIs). RFC 3987. Internet Engineering Task Force. <http://www.ietf.org/rfc/rfc3987.txt> (2005)
 29. Euzenat, J., Shvaiko, P.: *Ontology Matching*. Springer, Heidelberg (2007)
 30. Fielding, R.: *Architectural styles and the design of network-based software architectures*. Ph.D. thesis, University of California, Irvine (2000)
 31. Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., Berners-Lee, T.: Hypertext transfer protocol – HTTP/1.1. RFC 2616. Internet Engineering Task Force. <http://www.ietf.org/rfc/rfc2616.txt> (1999)
 32. Flouris, G., Fundulaki, I., Pediaditis, P., Theoharis, Y., Christophides, V.: Coloring RDF triples to capture provenance. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) *Proceedings of the Eighth International Semantic Web Conference (ISWC 2009)*, Chantilly. Lecture Notes in Computer Science, vol. 5823, pp. 196–212. Springer, Berlin (2009)
 33. Golbeck, J.: *Computing and applying trust in web-based social networks*. Ph.D. thesis, University of Maryland (2005)
 34. W. O. W. Group: Owl 2 web ontology language document overview, W3C Recommendation, World Wide Web Consortium (W3C). <http://www.w3.org/TR/owl2-overview/> (Oct 2009)
 35. Halpin, H., Presutti, V.: An ontology of resources for linked data. In: *Proceedings of the Linked Data on the Web (LDOW 2009)*, Madrid, co-located with the 18th International World Wide Web Conference (WWW 2009) (2009)
 36. Harth, A.: VisiNav: visual web data search and navigation. In: *Proceedings of the 20th International Conference on Database and Expert Systems Applications (DEXA 2009)*, Linz, pp. 214–228 (2009)
 37. Harth, A., Hose, K., Karnstedt, M., Polleres, A., Sattler, K.-U., Umbrich, J.: Data summaries for on-demand queries over linked data. In: *Proceedings of the 19th International Conference on World Wide Web (WWW 2010)*, Raleigh, pp. 411–420. ACM, New York (2010)
 38. Harth, A., Umbrich, J., Decker, S.: Multicrawler: a pipelined architecture for crawling and indexing semantic web data. In: *Proceedings of the Fifth International Semantic Web Conference (ISWC 2006)*, Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 258–271. Springer, Berlin (2006)
 39. Hawke, S.: Rule interchange format (RIF) HCLS 2009 – use case: vocabulary mapping. <http://www.w3.org/2009/Talks/0903-rif/Overview.html> (2009)
 40. Heimbigner, D., McLeod, D.: A federated architecture for information management. *ACM Trans. Inf. Syst.* **3**(3), 253–278 (1985)
 41. Heinsohn, J., Kudenko, D., Nebel, B., Profitlich, H.-J.: An empirical analysis of terminological representation systems. *Artif. Intell.* **68**(2), 367–397 (1994)
 42. Heitmann, B., Kinsella, S., Hayes, C., Decker, S.: Implementing semantic web applications: reference architecture and challenges. In: *Proceedings of Fifth International Workshop on Semantic Web Enabled Software Engineering (SWESE 2009)*, Virginia, co-located with Eighth International Semantic Web Conference (ISWC 2009), Chantilly (2010)
 43. Hogan, A., Harth, A., Passant, A., Decker, S., Polleres, A.: Weaving the Pedantic Web. In: *Proceedings of the International Workshop on Linked Data on the Web (LDOW 2010)*, Raleigh, co-located with the 19th International World Wide Web Conference (WWW 2010) (2010)
 44. Hogan, A., Harth, A., Umbrich, J., Decker, S.: Towards a scalable search and query engine for the web. In: *Proceedings of the 16th International Conference on World Wide Web (WWW 2007)*, Banff, pp. 1301–1302 (2007)
 45. Huynh, D.F., Karger, D.R., Miller, R.C.: Exhibit: lightweight structured data publishing. In: *Proceedings of the 16th International Conference on World Wide Web (WWW 2007)*, Banff, pp. 737–746. ACM, New York (2007)
 46. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The EigenTrust algorithm for reputation management in P2P networks. In: *Proceedings of the 12th International World Wide Web Conference (WWW 2003)*, Budapest (2003)
 47. Kaufmann, E., Bernstein, A.: How useful are natural language interfaces to the semantic web for casual end-users? In: *Proceedings of Sixth International Semantic Web Conference (ISWC 2007)*,

- Busan. Lecture Notes in Computer Science, vol. 4825, pp. 281–294. Springer, Berlin (2007)
48. Kjernsmo, K., Passant, A.: SPARQL new features and rationale, W3C Working Draft. <http://www.w3.org/TR/sparql-features/> (July 2009)
 49. Kobilarov, G., Dickinson, I.: Humboldt: exploring linked data. In: Proceedings of the Linked Data on the Web (LDOW 2008), co-located with the 17th International Conference on World Wide Web (WWW 2008), Beijing (2008)
 50. Kossmann, D.: The state of the art in distributed query processing. *ACM Comput. Surv.* **32**(4), 422–469 (2000)
 51. Krötzsch, M., Vrandečić, D., Völkel, M.: Semantic MediaWiki. In: Proceedings of the Fifth International Semantic Web Conference (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 935–942. Springer, Berlin (2006)
 52. Maedche, A., Staab, S., Stojanovic, N., Studer, R., Sure, Y.: Semantic portal – the seal approach. In: Fensel, D., Hendler, J.A., Lieberman, H., Wahlster, W. (eds.) *Spinning the Semantic Web*, pp. 317–359. MIT Press, Cambridge. <http://www.bibsonomy.org/bibtex/2b138c3a15839372089ee34141592aab2/erispublications> (2003)
 53. McBride, B.: Jena: implementing the RDF model and syntax specification. In: Proceedings of the Second International Workshop on the Semantic Web (Sem Web 2001), Honkong (2001)
 54. Moreau, L., Freire, J., Futrelle, J., McGrath, R.E., Myers, J., Paulson, P.: The open provenance model: an overview. In: Proceedings of Second International Provenance and Annotation Workshop (IPAW 2008), Salt Lake City. Lecture Notes in Computer Science, vol. 5272, pp. 323–326. Springer, Berlin (2008)
 55. Motro, A.: Superviews: virtual integration of multiple databases. *IEEE Trans. Softw. Eng.* **SE-13**(7), 785–798 (July 1987)
 56. Neumann, T., Weikum, G.: The RDF-3x engine for scalable management of RDF data. *VLDB J.* **19**(1), 91–113 (2010)
 57. OAEI: Ontology Alignment Evaluation Initiative: <http://oaei.ontologymatching.org/>
 58. OpenID: OpenID foundation website. <http://openid.net/> (2010)
 59. Oren, E., Delbru, R., Catasta, M., Cyganiak, R., Stenzhorn, H., Tummarello, G.: Sindice.com: a document-oriented lookup index for open linked data. *Int. J. Metadata Semant. Ontol.* **3**(1), 37–52 (2008)
 60. Oren, E., Delbru, R., Decker, S.: Extending faceted navigation for RDF data. In: Proceedings of the Fifth International Semantic Web Conference (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 559–572. Springer, Berlin (2006)
 61. OWLIM: OWLIM semantic repository. <http://www.ontotext.com/owlim>
 62. Papakonstantinou, Y., Garcia-Molina, H., Widom, J.: Object exchange across heterogeneous information sources. In: Proceedings of the 11th International Conference on Data Engineering (ICDE 1995), Taipei, pp. 251–260. IEEE Computer Society, Washington, DC (1995)
 63. Pietriga, E., Bizer, C., Karger, D.R., Lee, R.: Fresnel: a browser-independent presentation vocabulary for RDF. In: Proceedings of the Fifth International Semantic Web Conference (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 158–171. Springer, Berlin (2006)
 64. Quan, D.A., Karger, R.: How to make a semantic web browser. In: WWW 2004: Proceedings of the 13th International Conference on World Wide Web (WWW 2004), New York, pp. 255–265. ACM, New York (2004)
 65. Quilitz, B., Leser, U.: Querying distributed RDF data sources with SPARQL. In: Proceedings of the Fifth European Semantic Web Conference (ESWC 2008), Tenerife. Lecture Notes in Computer Science, vol. 5021, pp. 524–538. Springer, Berlin (2008)
 66. Raimond, Y., Sutton, C., Sandler, M.: Interlinking music-related data on the web. *IEEE Multimedia* **16**, 52–63 (2009)
 67. Rescorla, E.: HTTP over TLS. RFC 2818. Internet Engineering Task Force (2000)
 68. Sagonas, K., Swift, T., Warren, D.S.: XSB as an efficient deductive database engine. *SIGMOD Rec.* **23**(2), 442–453 (1994)
 69. Schenk, S., Saathoff, C., Staab, S., Scherp, A.: Semaplorer – interactive semantic exploration of data and media based on a federated cloud infrastructure. *J. Web Semant.* **7**(4), 298–304 (2009)
 70. Schenk, S., Staab, S.: Networked graphs: a declarative mechanism for SPARQL rules, SPARQL views and RDF data integration on the web. In: Proceedings of the 17th International

- Conference on World Wide Web (WWW 2008), Beijing, pp. 585–594. ACM, New York (2008)
71. Schraefel, M., Golbeck, J., Degler, D., Bernstein, A., Rutledge, L.: Semantic web user interactions: exploring HCI challenges. In: Extended Abstracts on Human Factors in Computing Systems, (CHI 2008), Atlanta, pp. 3929–3932. ACM, New York (2008)
 72. Seaborne, A., Manjunath, G., Bizer, C., Breslin, J., Das, S., Davis, I., Harris, S., Idehen, K., Corby, O., Kjærnsmo, K., Nowack, B.: SPARQL update – a language for updating RDF graphs, W3C Member Submission. <http://www.w3.org/Submission/SPARQL-Update/> (July 2008)
 73. Selberg, E., Etzioni, O.: Multi-service search and comparison using the metacrawler. In: Proceedings of the Fourth International World Wide Web Conference (WWW 1995), Boston, pp. 195–208 (1995)
 74. SPARQL: SPARQL query language for RDF, W3C Recommendation. <http://www.w3.org/TR/rdf-sparql-query/> (Jan 2008)
 75. Stefaner, M., Ferr, S., Perugini, S., Koren, J., Zhang, Y.: User interface design. In: Dynamic Taxonomies and Faceted Search: Theory, Practice, and Experience. The Information Retrieval Series, vol. 25, pp. 75–112. Springer, Heidelberg (2009) (Chap. 4)
 76. Story, H., Harbulot, B., Jacobi, I., Jones, M.: FOAF+TLS: RESTful authentication for the Social Web. In: Proceedings of the First Workshop on Trust and Privacy on the Social and Semantic Web (SPOT 2009), Heraklion (2009)
 77. Stuckenschmidt, H., Vdovjak, R., Broekstra, J., Houben, G.-J.: Towards distributed processing of RDF path queries. *Int. J. Web Eng. Technol.* **2**(2–3), 207–230 (2005)
 78. Virtuoso: Virtuoso universal server. <http://virtuoso.openlinksw.com/>
 79. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Discovering and maintaining links on the web of data. In: Proceedings of the Eighth International Semantic Web Conference (ISWC 2009), Chantilly. Lecture Notes in Computer Science, vol. 5823, pp. 650–665. Springer, Berlin (2009)
 80. Wiederhold, G.: Mediators in the architecture of future information systems. *Computer* **25**(3), 38–49 (1992)
 81. Yee, K.-P., Swearingen, K., Li, K., Hearst, M.: Faceted metadata for image search and browsing. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2003), Ft. Lauderdale, pp. 401–408. ACM, New York (2003)



3 Semantic Annotations and Retrieval: Manual, Semiautomatic, and Automatic Generation

Kalina Bontcheva · Hamish Cunningham
University of Sheffield, Sheffield, UK

3.1	<i>Scientific and Technical Overview</i>	78
3.1.1	Encoding Semantic Annotations	81
3.1.2	Manual Semantic Annotation	82
3.1.3	Automatic and Semiautomatic Annotation	86
3.1.4	Entity Disambiguation	95
3.1.5	Annotation Retrieval	96
3.2	<i>Example Applications</i>	102
3.2.1	GATE: A Semantic Annotation Framework	102
3.2.2	Large-Scale Semantic Annotation of News	106
3.2.3	Large-Scale Semantic Patent Processing	108
3.3	<i>Future Issues</i>	111
3.4	<i>Cross-References</i>	113

Abstract: The semantic annotation of textual Web content is key for the success of the Semantic Web. This entry reviews key approaches and state-of-the-art systems, as well as drawing conclusions on outstanding challenges and future work.

First, the problem of semantic annotation is defined and distinguished from other related research fields. Manual annotation tools are discussed next in the context of key requirements, such as support for diverse document formats, multiple ontologies, and collaborative, Web-based annotation.

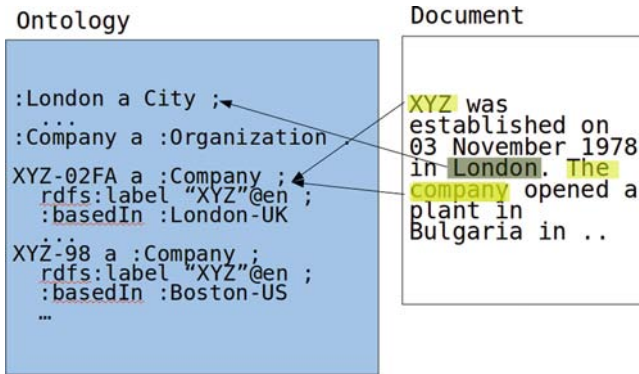
Next, the entry discusses ontology-oriented, semiautomatic, and automatic systems, which typically target ontologies as their output format, but do not use them as a knowledge resource during semantic analysis. Then a number of more advanced ontology-based semantic annotation approaches are presented and compared to one another. Particular emphasis is on scalability (i.e., the ability to process millions of documents) and customization (i.e., how easy it is to adapt these systems to new domains and/or ontologies).

The semantic retrieval of documents enables users to find all documents that mention one or more instances from the ontology and/or relations. The queries can also mix free-text keywords, not just the annotations. Here different types of retrieval tools are reviewed, some of which provide document browsing functionality as well as search refinement capabilities. The entry then provides in-depth examples of three semantic annotation applications: the GATE framework, News Collector, and large-scale patent processing. Future issues to be addressed are making use of linked data, dealing with large-scale, highly ambiguous ontologies, multilinguality, lexicalization of ontologies, and from an implementational perspective, semantic annotation as a service.

3.1 Scientific and Technical Overview

The Semantic Web is about adding a machine-tractable, repurposeable layer that complements the “traditional” Web of natural language hypertext. An important aspect of the World Wide Web is that it has been based largely on human-written materials, and in making the shift to the next-generation knowledge-based Web, human language will remain key. One particular example of the continuing importance of human language content on the Web comes from the success of Web 2.0 and social media. For instance, the growth in Twitter alone between 2008 and 2009 was over 1,000% and it is projected that by 2010, around 10% of all Internet users will be publishing content on Twitter. At the same time, there are over 70 million blogs and the average Facebook user has around 160 connections, many of whom are posting content in natural language on a daily basis. These users are also publishing other media online, such as photos and videos but these are outside the scope of this entry.

In the knowledge management context, Gartner reported (<http://www3.gartner.com/DisplayDocument?id=379859>) that more than 95% of human-to-computer information input involves textual language and this trend will remain stable. They also report that by 2012, taxonomic and hierarchical knowledge mapping and indexing will be prevalent in



■ Fig. 3.1

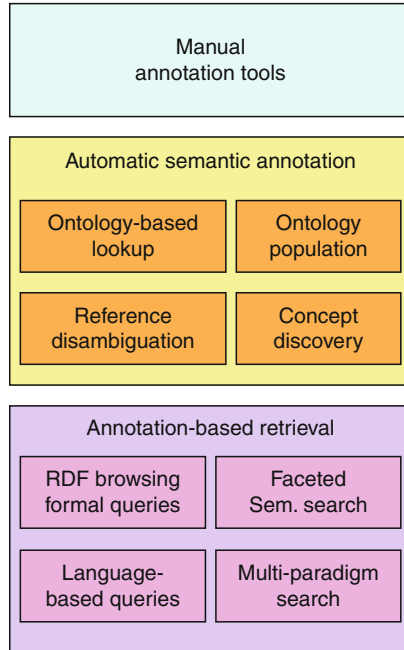
Semantic annotation example

almost all information-rich applications. There is a tension here: between the increasingly rich semantic models in Semantic Web systems on the one hand and the continuing prevalence of human language materials on the other (see [Ontologies and the Semantic Web](#), for an introduction to ontologies and the Semantic Web).

The process of tying semantic models and natural language together is referred to as *semantic annotation*. This process may be characterized as the dynamic creation of interrelationships between *ontologies* and unstructured and semi-structured documents in a bidirectional manner. From a technological perspective, semantic annotation is about annotating in texts all mentions of concepts from the ontology (i.e., classes, instances, properties, and relations), through metadata referring to their URIs in the ontology. Approaches that only enhance the ontology with new instances derived from the texts are typically referred to as *ontology population*.

A semantic annotation example is shown in [Fig. 3.1](#), where the strings “XYZ” and “the company” are marked as referring to the instance with URI XYZ-02FA, which is of class company. From an implementational perspective, the semantic annotation task is often broken down into two main phases: ontology-based lookup and reference disambiguation (see [Fig. 3.2](#)). Ontology-based lookup is concerned with identifying all candidate mentions of concepts from the ontology. In this example, there are two candidates that match the string XYZ, based on their RDF labels: XYZ-02FA and XYZ-98. The reference disambiguation step then uses contextual information from the text as well as knowledge from the ontology to disambiguate the mentions to the correct ontology concept. In this example, the text mentions London and in the ontology, XYZ-02FA is the candidate company established in London.

Some semantic annotation systems also perform ontology population (see [Fig. 3.2](#)), that is, in addition to annotating the documents with respect to an ontology, they also enrich the ontology itself with new instances not already present in the ontology. For example, if a new British prime minister comes to power and a system is annotating news documents, then it can discover the new prime minister’s name from the incoming



■ Fig. 3.2

Semantic annotation and retrieval tasks and approaches

articles. It must be noted that ontology population is a much harder task than ontology-based lookup and reference disambiguation, since it can introduce noisy, unreliable information in the ontology. An even more challenging problem is new concept discovery where a system can also learn new ontological classes and relations. The latter is typically carried out in a separate step, where domain experts can check the quality and validity of the newly discovered facts, prior to placing them in the ontology.

Semantic annotation can be performed manually, automatically, or semiautomatically, that is, first an automatic system creates some annotations and these are then post-edited and corrected by human annotators. Also, by definition all annotations are tied to one or more ontologies. Therefore, if an ontology changes or needs to be substituted by a different ontology, then all or some of the semantic annotation of the documents will need to be redone. Consequently, ontology evolution and the size of textual content on the Web make manual annotation infeasible in most cases, apart from very limited domains and applications. It is used primarily as means for checking the quality of the automatic methods, as well as for estimating the effort required for semiautomatic annotation.

Information Extraction (IE), a form of natural language analysis, is becoming a central technology for bridging the gap between unstructured text and formal knowledge expressed in ontologies. Ontology-Based IE (OBIE) is IE that is adapted specifically for the semantic annotation task. One of the important differences between traditional IE and OBIE is in the use of a formal ontology as one of the system's inputs and as the target

output. Some researchers (e.g., [1]) call ontology-based any system that specifies its outputs with respect to an ontology, however, in general, if a system only has a mapping between the IE outputs and the ontology, this is not sufficient and therefore, such systems should be referred as *ontology-oriented*.

Another distinguishing characteristic of the ontology-based IE process is that it not only finds the (most specific) class of the extracted entity, but also identifies it by linking it to its semantic description in the instance base, typically via a URI. This allows entities to be traced across documents and their descriptions to be enriched during the IE process. In practical terms, this requires automatic recognition of named entities, terms, and relations and also coreference resolution both within and across documents. These more complex algorithms are typically preceded by some linguistic preprocessing (tokenization, Part-Of-Speech (POS) tagging, etc.).

Irrespective of the techniques used, the semantic annotation of textual content enables *semantic-based document retrieval* (see [▶ Fig. 3.2](#)). This task is a modification of classical Information Retrieval (IR), but documents are retrieved on the basis of relevance to ontology concepts, as well as words. Nevertheless the basic assumption is quite similar – a document is characterized by the bag of tokens constituting its content, disregarding its structure. While the basic IR approach considers the word stems as tokens, there has been considerable effort for the last decade toward using word-senses or lexical concepts (see [2, 3]) for indexing and retrieval. The semantic annotations can be regarded as a special kind of token to be indexed and retrieved. With respect to techniques, work on semantic-based document retrieval is significantly less advanced than that on semantic annotation techniques, largely because the latter are enablers of the former. In addition, sufficiently scalable semantic repositories have only recently become available (for further details on semantic repositories see [▶ Storing the Semantic Web: Repositories](#)).

Semantic annotation is relevant in many application contexts, for example, knowledge management (see [▶ Knowledge Management in Large Organizations](#)), ([▶ eBusiness](#)), and ([▶ eScience](#)) (see the eponymous chapters in this handbook), and many large-scale implemented systems are already deployed and used on a daily basis.

3.1.1 Encoding Semantic Annotations

The first issue that needs to be addressed by any semantic annotation system is how to encode the annotations in documents. Some commonly used approaches are:

- As inline markup within the document's text content, with URIs pointing to the ontology (see [▶ Fig. 3.13](#))
- As RDF markup attached to the start/end of the document (see [▶ Fig. 3.3](#))
- As standoff RDF markup pointing to the document, but stored in a separate file and/or loaded within a semantic repository (see [▶ Fig. 3.10](#))

The trade-offs between these three approaches are several. First, representing each annotation inline, on each occurrence of the target instance/class has the advantage over

```

<!--ONTOMAT-ANNOTATION-BEGIN<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:iswc="http://annotation.semanticweb.org/2004/iswc#"
  ...
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
<owl:Ontology rdf:about="http://www.dcs.shef.ac.uk/~kalina/index.html#">
  <owl:imports rdf:resource="http://annotation.semanticweb.org/ontologies/cream/ontomat#">
  <owl:imports rdf:resource="http://annotation.semanticweb.org/2004/iswc#">
</owl:Ontology>
<iswc:Organization rdf:about="http://www.dcs.shef.ac.uk/~kalina/index.html#University_of_Sheffield">
  <rdfs:label> University of Sheffield</rdfs:label>
  <iswc:has_affiliate>
    <iswc:Person rdf:about="http://www.dcs.shef.ac.uk/~kalina/index.html#Kalina_Bontcheva">
      <iswc:phone>(+44 - 114) 222 1930</iswc:phone>
      <iswc:fax>(+44 - 114) 222 1810</iswc:fax>
      <rdfs:label>Kalina Bontcheva</rdfs:label>
    </iswc:Person>
  </iswc:has_affiliate>
</iswc:Organization>
...
</rdf:RDF>
-->

```

■ Fig. 3.3

Example RDF markup attached to a semantically annotated document

the other two representations when a system needs to retrieve the specific place(s) within documents where this class/instance is mentioned. The example shown in [Fig. 3.13](#) shows how all references to George Bush are annotated accordingly. In comparison, the other two representations only encode the fact that certain instances are mentioned in a particular document, but it is not possible to retrieve examples of where exactly these occur in the text. This approach makes the semantic annotation task considerably easier, since annotators only need to annotate only one of the mentions. Consequently, this also makes data storage and retrieval requirements smaller.

When comparing the second representation (RDF markup appended to the original document) to the third (RDF markup in a separate file), the choice depends on whether it is feasible to modify the document content itself. For an introduction to RDF, see

[Semantic Annotation and Retrieval: RDF](#).

3.1.2 Manual Semantic Annotation

Frameworks and user interface tools for manual semantic annotation need to address several challenges:

- First, as discussed above, they need to support references to ontology concepts via URIs.
- Second, given that manual annotation is time consuming, the tools should ideally be collaborative and also Web-based to enable distributed teams of annotators to share the work.

- Third, the tools need to go beyond annotation of classes and instances, and support also annotation of property and relation values.
- Fourth, the tools need to support annotation with respect to multiple ontologies and also scale well for large ontologies with many classes and relations. As discussed in [4], the annotation of relationships is significantly more time consuming for users and therefore a suitably supportive GUI is required.
- Last, but not least, manual annotation tools need to support multiple document formats, going beyond HTML toward PDF, XML, images (e.g., PNG, JPEG), and video.

Next, several state-of-the-art manual annotation tools are discussed in the context of these challenges. For a description of some older systems please refer to [5].

A comprehensive semantic annotation framework is CREAM [4]. It not only addresses the requirements listed above, but it also provides a document editor that supports the creation of semantic annotations as an integral part of document authoring. Another distinguishing feature is the RDF crawler, which collects relevant entities from already published Semantic Web RDF data and makes these available to the human annotators, so they can reuse already existing instances, instead of creating new ones. CREAM also allows for the integration of automatic tools to bootstrap the manual annotation process (discussed in the following section).

CREAM's manual annotation editor is OntoMat Annotizer (see [▶ Fig. 3.4](#)) and it runs in a Web browser. There is an ontology guidance and fact browser, which allows users to expand the ontology with new data, for example, add a new instance. Document-based annotation is carried out by selecting parts of the text and then dropping them on the desired ontology class or, once a class has been chosen, in the property template for that class, in order to instantiate property values (e.g., a person's name or their date of birth). The example in [▶ Fig. 3.4](#) shows a Web page annotated with people, projects, and organizations. However, although the new instances and annotations were created by first selecting them in the text, the mentions of these in the text itself are not highlighted, due to the fact that CREAM uses RDF triples, which are independent of the text itself (see [▶ Fig. 3.3](#)).

Another manual semantic annotation editor for Web pages, similar to OntoMat, is SMORE (<http://www.mindswap.org/2005/SMORE/>) (see [▶ Fig. 3.5](#)). It also integrates the SWOOP (<http://www.mindswap.org/2004/SWOOP/>) ontology editor. SMORE supports ontology navigation in order to select classes and properties and create triples to be added to the HTML pages. It also verifies the domain and range constraints on annotations to detect inconsistencies.

The W3C Annotea annotation framework and its extensions (e.g., [6]) support collaborative semantic annotation of documents accessible over the Internet, in multiple document formats, for example, HTML, PDF, images, and video. The framework uses RDF to model annotations as a set of statements. Annotations range from simple text comments, through hyperlinks, to controlled vocabulary statements (e.g., WordNet) and ontologies. As discussed in [6], annotations with respect to ontologies are modeled

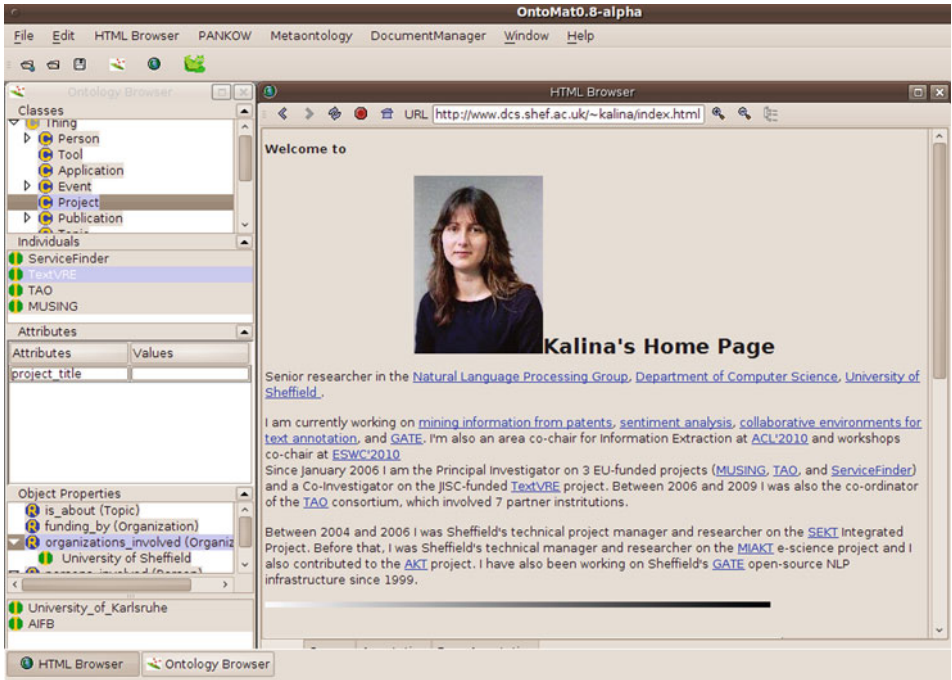


Fig. 3.4
OntoMat manual annotation tool

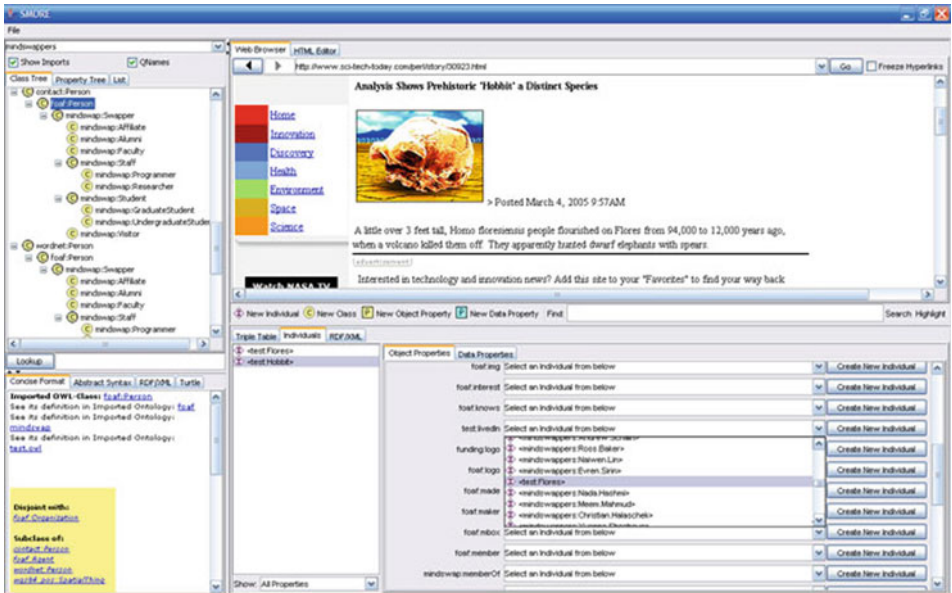




Fig. 3.5
SMORE manual annotation UI

through reification in order to support provenance, that is, information on who annotated what. The problem with reification, however, is that it is computationally expensive. The authors have therefore proposed to investigate named graphs in future work as a less expensive way to represent semantic annotations.

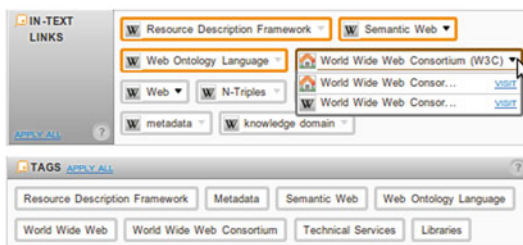
Zemanta (<http://www.zemanta.com>) is an online annotation tool for blog and e-mail content, which helps users insert tags and links through recommendations.  [Figure 3.6](#) shows an example text and the recommended tags, potential in-text link targets (e.g., the W3C Wikipedia article and the W3C home page), and other relevant articles. It is then for the user to decide which of the tags should apply and which in-text link targets they wish to add. In this example, in-text links have been added for the terms highlighted in orange, all pointing to the Wikipedia articles on the respective topics.

There are also a number of multimedia semantic annotation tools, which are covered in more detail in  [Multimedia, Broadcasting, and eCulture](#). To take just one example, PhotoStuff [7] is an image annotation tool, which supports semantic annotation of photos and regions of images with respect to OWL and RDFS ontologies. It defines an image region ontology (<http://www.mindswap.org/2005/owl/digital-media>), which has a set of useful concepts for image annotation. The semantic annotation interface is based

Your content enhanced!

The term was coined by **World Wide Web Consortium (W3C)** director Tim Berners-Lee. According to the original vision, the availability of machine-readable metadata would enable automated agents and other software to access the Web more intelligently. The agents would be able to perform tasks automatically and locate related information on behalf of the user.

While the term "Semantic Web" is not formally defined it is mainly used to describe the model and technologies proposed by the W3C. These technologies include the **Resource Description Framework (RDF)**, a variety of data interchange formats (e.g., RDF/XML, N3, Turtle, N-Triples), and notations such as RDF Schema (RDFS) and the **Web Ontology Language (OWL)**, all of which are intended to provide a formal description of concepts, terms, and relationships within a given knowledge domain.



Content recommendations



 **Fig. 3.6**

Zemanta's online tagging demo

on forms, which provide slots for all properties of the chosen class and the user can then specify the values. For example, if the astronaut class is chosen for a given part of an image, then the form is populated with properties such as date of birth, education, and employer. The manual RDF annotations can then be uploaded into a semantic portal where they are published and made available for semantic searches (see ▶ Sect. 3.1.5). Provenance in this case is modeled at a file level, rather than annotation level, that is, each RDF file with semantic annotations is tagged with its creator name, description, and time stamp. On the one hand, this is a far more coarse-grained provenance model, but on the other, it is more computationally efficient.

In summary, while manual semantic annotation can be feasible in limited domains or through involving multiple annotators over the Web, it is in general considered too expensive to carry out without any automation. Consequently, the next section introduces semiautomatic and automatic approaches, many of which have been combined already with manual annotation.

3.1.3 Automatic and Semiautomatic Annotation

As discussed in the introduction, there are a number of ontology-oriented semantic annotation systems, which, unlike ontology-based ones, do not incorporate ontologies into the semantic analysis, but either use them as a bridge between the linguistic output and the final annotation (as with AeroDAML) or rely on the user to provide the relevant information through manual annotation (as with the Amilcare-based tools).

Information Extraction (IE) is one of the most commonly used techniques for (semi-) automatic semantic annotation. For example, when annotating information about companies, key information to be identified would be the company address, contact phone, fax numbers, and e-mail address, products and services, members of the board of directors, and so on. The field of information extraction has been driven by two major US international evaluation programs, from 1987 until 1997, the Message Understanding Conferences [8, 9] and since 2000, the Automatic Content Extraction Evaluation (ACE) [10].

The main tasks carried out during information extraction are:

- Named entity recognition, which consists of the identification and classification of different types of names in text
- Coreference resolution, which is the task of deciding if two linguistic expressions refer to the same entity in the discourse
- Relation extraction, which identifies relations between entities in text

Information extraction usually employs the following natural language processing components: Part-Of-Speech (POS) taggers, morphological analyzer, named entity recognizers, full (or shallow) parsing, and semantic interpretation. These linguistic processors are generally available (e.g., in language processing frameworks such as

GATE [11]), although some may require domain adaptation. For example, while a Parts-Of-Speech tagger can be reused mostly as is, a named entity recognizer would usually need adaptation to new application domains.

There are two main classes of approaches to information extraction:

1. Rule-based systems which are built by language engineers, who design lexicons and rules for extraction.
2. Machine-learning systems that are trained to perform one or more of the IE tasks. Learning systems are given either an annotated training corpus (i.e., supervised machine learning) or unannotated corpus together with a small number of seed examples (i.e., unsupervised or lightly supervised methods).

The advantages of rule-based approaches are that they do not require training data to create (although a small gold standard is needed for evaluation) and harness human intuition and domain knowledge. Depending on the lexical and syntactic regularity of the target domain, rule creation ranges from extremely fast (when few, clear patterns exist) to rather time consuming (if more ambiguities are present). Depending on the system design, some changes in requirements may be hard to accommodate. Since rule-based systems tend to require at least basic language processing skills, they are sometimes perceived as more expensive to create.

In comparison, machine-learning approaches typically require at least some human-annotated training data in order to reach good accuracy. While the cost per individual annotator is lower than the cost of language engineers, given the size of data needed, often more than one or two annotators are required. This raises the problem of inter-annotator agreement (or consistency), since the accuracy of the learnt models can be affected significantly by noisy, contradictory training data. However, getting training annotators to agree on their labels is again dependent on the complexity of the target annotations and could in itself be rather time consuming. Another potential problem could arise if the semantic annotation requirements change after the training data has been annotated, since this may require substantial re-annotation.

To summarize, both types of approaches have advantages and drawbacks and the choice of which one is more appropriate for a given application depends on the target domain, the complexity of the semantic annotations (including the size of the ontology), and the availability of trained human annotators and/or language engineers. Last but not least, there is no reason why one cannot have a hybrid approach, which uses both rules and machine learning.

Next, some representative semantic annotation systems are discussed with emphasis on their extraction components.

AeroDAML [12] is an annotation tool created by Lockheed Martin that applies IE techniques to automatically generate DAML annotations from Web pages. The aim is to provide naive users with a simple tool to create basic annotations without having to learn about ontologies, in order to reduce time and effort and to encourage people to semantically annotate their documents. AeroDAML links most proper nouns and common types of relations with classes and properties in a DAML ontology.

There are two versions of the tool: a Web-enabled version that uses a default generic ontology, and a client-server version that supports customized ontologies. In both cases, the user enters a URI (for the former) and a filename (for the latter) and the system returns the DAML annotation for the Web page or document. It provides a drag-and-drop tool to create static (manual) ontology mappings, and also includes some mappings to predefined ontologies.

AeroDAML consists of the AeroText IE system, together with components for DAML generation. A default ontology that directly correlates to the linguistic knowledge base used by the extraction process is used to translate the extraction results into a corresponding RDF model that uses the DAML + OIL syntax. This RDF model is then serialized to produce the final DAML annotation. The AeroDAML ontology is comprised of two layers: a base layer comprising the common knowledge base of AeroText, and an upper layer based on WordNet [13]. AeroDAML can generate annotations consisting of instances of classes such as common nouns and proper nouns, and properties, of types such as coreference, Organization to Location, Person to Organization.

Amilcare [14] is an adaptive IE system that has been integrated in several different annotation tools for the Semantic Web. It uses machine learning (ML) to learn to adapt to new domains and applications using only a set of annotated texts (training data). It has been adapted for use in the Semantic Web by simply monitoring the kinds of annotations produced by the user in training, and learning how to reproduce them. The traditional version of Amilcare adds XML annotations to documents (inline markup); the Semantic Web version (used by Melita – see below) leaves the original text unchanged and produces the extracted information as triples of the form $\langle \text{annotation, startPosition, endPosition} \rangle$ (standoff markup – see [Sect. 3.1.1](#)). This means that it is left to the annotation tool and not the IE system to decide on the format of the ultimate annotations produced.

In the Semantic Web version, no knowledge of IE is necessary; the user must simply define a set of annotations, which may be organized as an ontology where annotations are associated with concepts and relations. The user then manually annotates the text using some interface connected to Amilcare, as described in the following systems. Amilcare works by preprocessing the texts using GATE's IE system ANNIE [15], and then uses a supervised machine learning algorithm [16] to induce rules from the training data.

Melita [17] is an ontology-based tool for semantic annotation, which provides a mechanism for a user to interact with an IE system (Amilcare). It consists of two main parts: an ontology viewer and a document editor. The two most interesting features of Melita are that it enables the user to tune the IE system to provide different levels of proactivity, and to schedule texts to provide timeliness (i.e., learning with minimum delay). The annotation cycle follows two phases: manual annotation (training of the system) and active annotation (where the system takes over the annotation automatically). At some point, the system will start suggesting annotations to the user (active annotation) and the user can correct these as necessary. The system can suggest annotations as either reliable or unreliable, depending on its confidence level about that

annotation. Reliable annotations need to be explicitly removed by the user, while unreliable annotations need to be explicitly added.

MnM [18] is a semantic annotation tool that provides support for annotating Web pages with semantic metadata. This support is semiautomatic, in that the user must provide some initial training information by manually annotating documents before the IE system (Amilcare) can take over. It integrates a Web browser, an ontology editor, and tools for IE, and has been described as “an early example of next-generation ontology editors” [18], because it is Web-based and provides facilities for large-scale semantic annotation of Web pages.

The philosophy behind MnM is that the semantic annotation of Web pages can, and should, be carried out by users without specialist skills in either language technology or knowledge engineering. It therefore aims to provide a simple system to perform knowledge extraction tasks at a semiautomatic level.

The five main steps to the underlying procedure are:

- The user browses the Web.
- The user manually annotates his chosen Web pages.
- The system learns annotation rules.
- The system tests the rules learnt.
- The system takes over automatic annotation, and populates ontologies with the instances found.

The ontology population process is semiautomatic and may require intervention from the user. First, it only deals with a predefined set of concepts in the ontology. Second, the system is not perfect and may miss instances in the text, or allocate them wrongly. Retraining can be carried out at any stage, however.

S-CREAM (Semiautomatic CREation of Metadata) [19] is a tool that provides a mechanism for automatically annotating texts, given a set of training data, which must be manually created by the user. It uses a combination of two tools: *Onto-O-Mat*, a manual annotation tool that implements the CREAM framework for creating relational metadata [20], and *Amilcare*.

As with the other *Amilcare*-based tools, *S-CREAM* is trainable for different domains, provided that the user creates the necessary training data. It essentially works by aligning conceptual markup (which defines relational metadata) provided by *OntoMat* with semantic markup provided by *Amilcare*. This problem is not trivial because the two representations may be very different. Relational metadata may provide information about relationships between instances of classes, for example that a certain hotel is located in a certain city. *S-CREAM* thus supports metadata creation with the help of a traditional IE system, and also provides other functionalities such as a Web crawler, a document management system, and a meta-ontology.

Wrapper-based Data Extraction for Ontology Population: *Lixto* [21] is a set of tools for writing wrappers that scrape Web pages and perform data extraction. As part of the *REVERSE* project these were used to build essentially an ontology population tool, which scrapes and syndicates information from publication pages. The output is an ontology of

researchers and publications that is populated automatically from the Web pages. However, unlike all previous systems, the goal here is only ontology population, that is, the original Web content is not annotated semantically.

Drawbacks of the Ontology-Oriented Approaches: One of the problems with ontology-oriented annotation tools, such as those reviewed here, is that they do not provide the user with a way to customize the integrated language processing directly. While many users would not need or want such customization facilities, users who already have ontologies with rich instance data will benefit if they can make these data available to the IE components. However, this is not possible when traditional IE methods such as Amilcare are used, because they are not aware of the existence of the user ontology.

The more serious problem however, as discussed in the S-CREAM system [19], is that there is often a gap between the IE output annotations and the classes and properties in the user's ontology. The solution proposed by the developers was to write logical rules to resolve this. For example, an IE system would typically annotate London and UK as locations, but extra rules are needed to specify that there is a containment relationship between the two. However, rule writing of this kind is too difficult for most users and therefore ontology-based semantic annotation algorithms were developed, as they annotate directly with the classes and instances from the user's ontology.

Magpie [22] is a suite of tools that supports the interpretation of Web pages and “collaborative sense-making.” It annotates Web pages with metadata in a fully automatic fashion and needs no manual intervention by matching the text against instances in the ontology (see [▶ Fig. 3.7](#)). It automatically populates an ontology from relevant Web sources, and can be used with different ontologies. The principle behind it is that it uses an ontology to provide a very specific and personalized viewpoint of the Web pages the user wishes to browse. This is important because different users often have different degrees of knowledge and/or familiarity with the information presented, and have different browsing needs and objectives.

Another interesting aspect of *Magpie* is that it maintains a kind of “browsing history” in windows called *collectors*. Each collector shows the instances of a given concept that have been mentioned on the page or a list of related instances (e.g., people working on a given project, which were not mentioned in this page). The user can then click on these instances and browse their semantic data or create *semantic bookmarks* to retrieve this information later through semantic queries.

However, *Magpie* relies on a prespecified ontology, which makes the system domain-dependent. *PowerMagpie* [23] is an extension of the approach, so that it identifies automatically, at runtime, the most appropriate ontology to be used for annotation. *PowerMagpie* displays two panels. The first one is called “Entities” and lists key terms from the Web page, as well as the ontological entities they refer to and navigation to the places where they are mentioned in the text. The second panel is called “Ontologies” and displays the automatically found ontologies, which were deemed relevant to the given page.

In *PowerMagpie*, semantic annotation is performed first by identifying statistically the domain terms and then matching them up against candidate ontologies. The system first

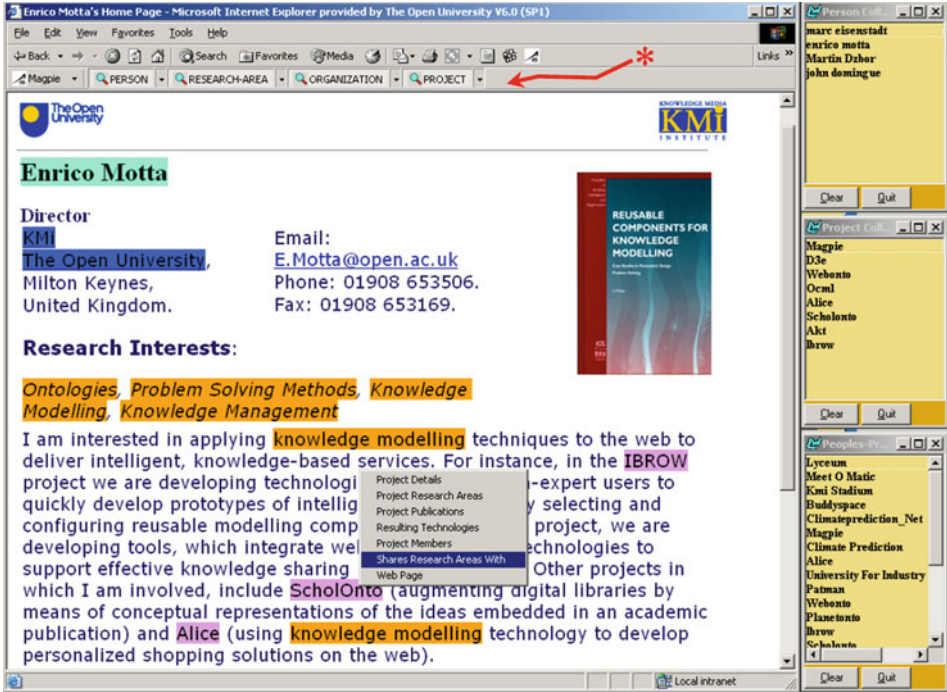


Fig. 3.7

Magpie example

used a TF*IDF term recognizer, but due to over-generation it was replaced by a call to the Yahoo! Term extraction service (<http://developer.yahoo.com/search/content/V1/termExtraction.html>). The key terms are then used to select dynamically one or more relevant ontologies. The matching is done on the basis of string similarity between the key terms and the labels of classes, instances, and properties in the ontology. Complex terms, for example, University of Sheffield, are matched as a whole, rather than matching university and Sheffield separately. However, it remains unclear how PowerMagpie would deal with named entity recognition and also with relation annotation.

PANKOW and OntoSyphon: The PANKOW system (Pattern-based Annotation through Knowledge on the Web) [24] exploits surface patterns and the redundancy on the Web to categorize automatically instances from text with respect to a given ontology. The patterns are phrases like: the < INSTANCE > < CONCEPT > (e.g., the Ritz hotel) and < INSTANCE > is a < CONCEPT > (e.g., Novotel is a hotel). The system constructs patterns by identifying all proper names in the text (using a Part-of-Speech tagger) and combining each one of them with each of the 58 concepts from their tourism ontology into a hypothesis. Each hypothesis is then checked against the Web via Google queries and the number of hits is used as a measure of the likelihood of this pattern being correct.


The system's best performance on this task in fully automatic mode is 24.9%, while the human performance is 62.09%. However, when the system is used in semiautomatic

mode, that is, it suggests the top five most likely concepts and the user chooses among them, then the performance goes up to 49.56%.

The advantages of this approach are that it does not require any text processing (apart from POS tagging) or any training data. All the information comes from the Web. However, this is also a major disadvantage because the method does not compare the context in which the proper name occurs in the document to the contexts in which it occurs on the Web, thus making it hard to classify instances with the same name that belong to different classes in different contexts (e.g., Niger can be a river, state, country, etc.). On the other hand, while IE systems are more costly to set up, they can take context into account when classifying proper names.

Another system similar to PANKOW is OntoSyphon [1], which uses the ontology as the starting point in order to carry out Web mining to populate the ontology with instances. It uses the ontology structure to determine the relevance of the candidate instances. However, it does not carry out semantic annotation of documents as such.

There has also been work on populating ontologies specifically from tabular data from the Web, for example, the AllRight system [25]. The approach is based on clustering, table identification, and conflict detection.

Open Calais is a commercial Web service provided by Thomson Reuters that carries out semantic annotation. At the time of writing, the target entities are mostly locations, companies, people, addresses, contact numbers, products, movies, etc. The events and facts extracted are those involving the above entities, for example, acquisition, alliance, and company competitor. The Calais OWL ontology is available at: <http://www.opencalais.com/documentation/opencalais-web-service-api/calais-ontology-owl>.  *Figure 3.8* shows an example text annotated with some entities.

The Calais service also carries out limited entity disambiguation for companies (with respect to a proprietary database of public companies); locations (using Freebase); and electronics (using Shopping.com).

The entity annotations include URIs, which allow access via HTTP to obtain further information on that entity via linked data. Currently OpenCalais links to eight linked datasets, including DBPedia, Wikipedia, IMDB, and Shopping.com. These broadly correspond to the entity types covered by the ontology.

The main limitation of Calais comes from its proprietary nature, that is, users send documents to be annotated by the Web service and receive results back, but they do not have the means to give Calais a different ontology to annotate with or to customize the way in which entity extraction works.

SemTag [26] performs large-scale semantic annotation with respect to the TAP ontology (<http://tap.stanford.edu/data/>). It first performs a lookup phase annotating all possible mentions of instances from the TAP ontology. In the second, disambiguation phase, SemTag uses a vector-space model to assign the correct ontological class or to determine that this mention does not correspond to a class in TAP. The disambiguation is carried out by comparing the context of the current mention against the contexts of instances in TAP with compatible aliases, using a window of ten words either side of the mention.

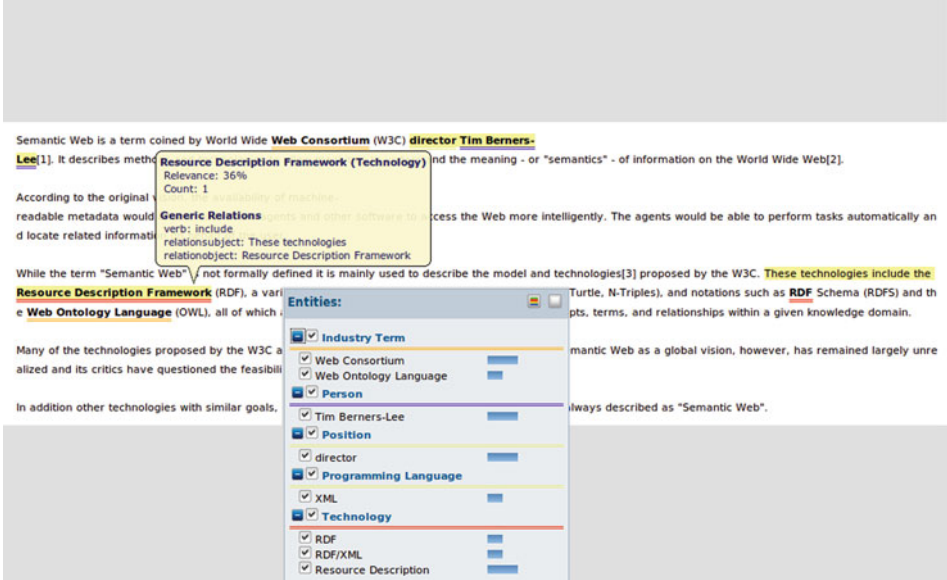


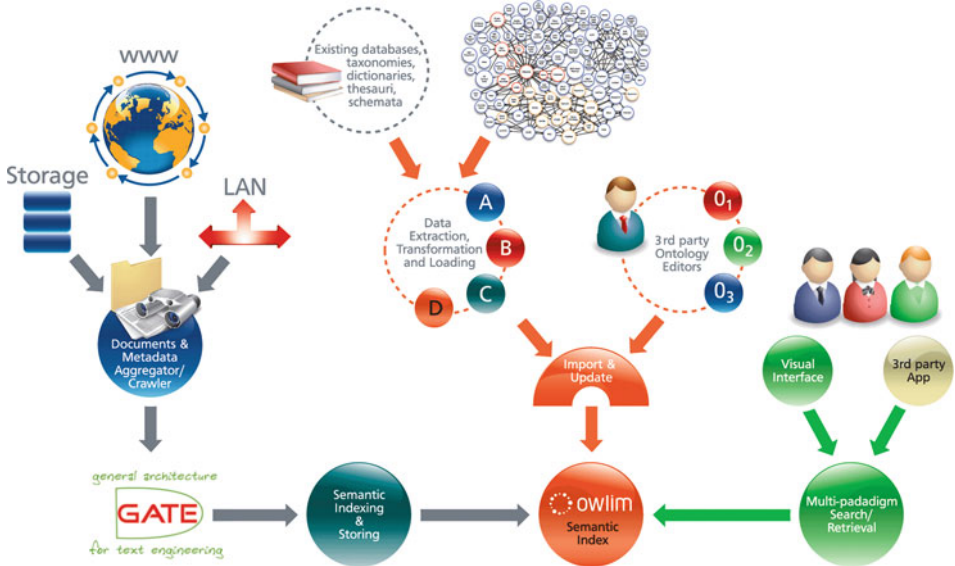
Fig. 3.8
Calais results on part of the Semantic Web Wikipedia entry

The TAP ontology, which contains about 65,000 instances, is very similar in size and structure to the KIM Ontology and KB (e.g., each instance has a number of lexical aliases). One important characteristic of both ontologies is that they are lightweight and encode only essential properties of concepts and instances. In other words, the goal is to cover frequent, commonly known and searched for instances (e.g., capital cities, names of presidents), rather than to encode an extensive set of axioms enabling deep, Cyc-style reasoning. As reported in [27], the heavyweight logical approach undertaken in Cyc is not appropriate for many NLP tasks.

The SemTag system is based on a high-performance parallel architecture – Seeker, where each node annotates about 200 documents per second. The demand for such parallelism comes from the big volumes of data that need to be dealt with in many applications and make automatic semantic annotation, the only feasible option. A parallel architecture of a similar kind is currently under development for KIM and, in general, it is an important ingredient of large-scale automatic annotation approaches.

The *KIM Semantic Annotation Platform* [28, 29] is an extendable platform for knowledge management that offers facilities for metadata creation, storage, and semantic-based search (see Fig. 3.9 for details). It also includes a set of front ends for online use that offer multi-paradigm search and semantically enhanced browsing (see Sect. 3.1.5).

KIM uses the PROTON ontology (<http://proton.semanticweb.org/>) that contains around 250 classes and 100 properties. The classes cover entities (such as people, organizations, locations, products) and events. The core entities addressed are roughly equivalent to those covered by OpenCalais.



■ Fig. 3.9

KIM architecture

The information extraction in KIM is based on the GATE framework [11]. The essence of KIM's semantic annotation is the recognition of named entities with respect to the KIM ontology. The entity instances all bear unique identifiers that allow annotations to be linked both to the entity type and to the exact individual in the instance base. For new (previously unknown) entities, new identifiers are allocated and assigned; then minimal descriptions are added to the semantic repository. The annotations are kept separately from the content, and an API for their management is provided.

The instance base of KIM is pre-populated with 200,000 entities of general importance that occur frequently in documents. The majority are different kinds of locations: continents, countries, cities, etc. Each location has geographic coordinates and several aliases (usually including English, French, Spanish, and sometimes the local transcription of the location name) as well as co-positioning relations (e.g., *subRegionOf*.) As previously shown by [30], IE systems need such data, because locations are difficult to recognize otherwise.

The difference between SemTag's TAP ontology and the KIM instance base is in the level of ambiguity. TAP has few entities sharing the same alias, while KIM has a lot more, due to its richer collection of locations.

At a conceptual level, KIM and SemTag differ significantly in their goal. Namely, SemTag aims only at accurate classification of the mentions that were found by matching the lexicalizations in the ontology. KIM, on the other hand, also aims to find *all mentions*, that is, coverage, as well as accuracy. The latter is a harder task because there tends to be a trade-off between accuracy and coverage. In addition, SemTag does not attempt to

discover and classify new instances, which are not already in the TAP ontology. In other words, KIM performs two tasks, ontology population with new instances and semantic annotation, while SemTag performs only semantic annotation.

KIM can also use linked data ontologies for semantic annotation. At present it has been tested with DBPedia, Geonames, WordNet, Musicbrainz, Freebase, UMBEL, Lingvoj, and the CIA World Factbook (for further details on linked data see [Semantic Annotation and Retrieval: Web of Data](#)). Those datasets are preprocessed and loaded to form an integrated dataset of about 1.2 billion explicit statements. Forward-chaining is performed to materialize another 0.8 billion implicit statements, in accordance with the semantics of the ontologies used in the datasets.

Another important issue is extensibility, where not only the ontology can be replaced or extended, but also there is the option to customize or replace the semantic annotation application used within KIM. This can be any GATE-based semantic application, including machine learning, rule-based, or any other text-processing component integrated in GATE. The semantic annotator could also be provided by any other system, provided that it is wrapped and plugged into KIM via its API. In fact, KIM's extensibility with respect to new ontologies and semantic annotators is what makes it more powerful than ready-made services, such as OpenCalais.

Ontology-Based Semantic Annotation via Hierarchical Learning: The Hieron system [31] implements a hierarchical learning approach for semantic annotation, which uses the target ontology as an essential part of the annotation process, by taking into account the relations between concepts and instances in the ontology.

As discussed above, conventional IE uses labels that have no specific relation among each other, that is, they are treated as independent by the learning algorithms (e.g., Person, Location). However, concepts in an ontology are related to each other (at the very least through the subsumption hierarchy) and therefore it is beneficial to feed this knowledge into the OBIE algorithms. The Hieron system has explored two aspects of using the ontology structure for semantic annotation. First, it derives ontology-induced measures, which are then used by the learning algorithm to evaluate how well it is learning to annotate the target concepts. Second, the authors introduce the Perceptron-based learning algorithm Hieron, which has a mechanism to handle effectively hierarchical classification, as is required for semantic annotation.

The approach was evaluated on a corpus of 290 news articles annotated manually with respect to an ontology of 146 classes. The results demonstrate clearly the benefits of using knowledge from the ontology as input to the information extraction process.

3.1.4 Entity Disambiguation

Gruhl et al. [32] focus in particular on the disambiguation element of semantic annotation and examine the problem of dealing with highly ambiguous cases. Their approach first restricts the part of the ontology used for producing the candidates, in this case by filtering out all information about music artists not mentioned in the given text. Second,

they apply lightweight language processing, such as POS tagging, and then use this information as input to a support vector machine classifier, which disambiguates on the basis of this information. The approach has been tested with the MusicBrainz ontology and a corpus of MySpace posts for three artists. While the ontology is very large (thus generating a lot of ambiguity), the texts are quite focused, which allows the system to achieve good performance. As discussed by the authors themselves, the processing of less focused texts, for example, Twitter messages or news articles, is likely to prove far more challenging.

The problem of person name disambiguation is a specific case of entity disambiguation, which has received attention in earlier work. For example, Aswani et al. [33] experimented with disambiguating author names in citations by using both contextual information (e.g., coauthor names) and additional evidence gathered from the Web, in combination with a similarity measure. Similar to Gruhl et al., the reported accuracy was very good, but again, the question remains open as to how well the approach will deal with other domains or text types.

Another approach to named entity disambiguation, called IdentityRank [34], was proposed in the context of the NEWS project. It tackles the problem of disambiguating named entities occurring in newspaper articles with respect to a news domain ontology. The algorithm exploits the metadata provided by news agencies, automatically detected named entities, and NewsCodes subject categories. It also takes into account the frequency of occurrence of entities in the last few days, as well as the frequency of occurrence in news with a given category. However, similar to the previously discussed approaches, IdentityRank has not been tested on other domains and applications and, thus, the general applicability of the approach outside the news domain remains unproven.

The IdRF framework for identity resolution [35] differs from the above work, since it exploits instead knowledge from the ontology, in order to determine whether a candidate mention from the text refers to a known instance in the ontology or a new one needs to be created. For efficiency reasons, the resolution process is divided into several steps. First is pre-filtering, which filters out the irrelevant parts of the ontology and forms a set of candidate entities. Next is the similarity measure stage, during which context from the text is compared against knowledge in the ontology to help with disambiguation. The last stage is called data integration and it determines whether to assert a new instance in the ontology or match the mention to an existing one, based on the similarity measures from the previous step. The potential drawback of this approach is that it requires the manual definition of the similarity metrics and pre-filtering criteria, which might prove complex as the size of the ontology grows.

3.1.5 Annotation Retrieval

Semantic annotations in documents enable users to find all documents that mention one or more instances from the ontology and/or relations. The queries can also mix free-text keywords, not just the annotations. Most retrieval tools provide also document browsing

functionality as well as search refinement capabilities. Due to the fact that documents can have hundreds of annotations (especially if every concept mention in the document is annotated), annotation retrieval on a large document collection is a very challenging task.

Annotation-based search and retrieval is different from traditional information retrieval, because of the underlying graph representation of annotations, which encode structured information about text ranges within the document. The encoded information is different from the words and inter-document link models used by Google and other search engines. In the case of semantic annotations, the case becomes even more complex, since they also refer to ontologies via URIs. While augmented full-text indexes can help with efficient access, the data storage requirements can grow exponentially with the cardinality of the annotation sets. Therefore different, more optimized solutions have been investigated.

The main difference from Semantic Web search engines, such as Swoogle [36], is the focus on annotations and using those to find documents, rather than forming queries against ontologies or navigating ontological structures. Similarly, semantic-based facet search and browse interfaces, such as /facet [37], tend to be ontology-based, whereas annotation-based facet interfaces (see KIM below) tend to hide the ontology and instead resemble more closely “traditional” string-based faceted search.

Next, several representative approaches are discussed.

Browsing RDF Annotations: The MINDSWAP SemPortal (<http://www.mindswap.org/>) publishes RDF annotations on the Web, for example, those created by PhotoStuff [7]. As can be seen in [▶ Fig. 3.10](#), the user can then browse these RDF instances via the portal and see any associated images and other documents. The provenance of the information is shown as a tooltip as shown in the example.

Natural Language Interfaces: These allow users to perform retrieval tasks using written or spoken language (e.g., English). The majority of work on natural language interfaces for

STS-2-Crew

Depictions:




STS-2-Crew	type	Shuttle Crew	
	depiction	http://www.spacefacts.de/mission/photo/sts-2.jpg	
	has pilot	Richard Truly	Submitters: Mike Grove. Files: http://SemSpace.mindswap.org/2004/submit-rdf/407.rdf
	has commander	Joseph Engle	
http://www.spacefacts.de/mission/photo/sts-2.jpg	depicts	STS-2-Crew	
STS2	has crew		

<http://semSPACE.mindswap.org/rdf/instance/?resource=http://www.spacefacts.de/mission/photo/sts-2.jpg>


▶ Fig. 3.10

SemPortal: Instance browsing example

the Semantic Web has focused on the problem of querying ontologies (e.g., [38–40]) or ontology authoring (e.g., [41, 42]). Using language-based queries for retrieving semantic annotations and associated documents is a somewhat different task, since the queries need to go beyond the ontology and into documents as well.

The QuestIO system [43], for example, has an ontology modeling document and the semantic annotations in them and uses it to help naive users to search through RDF annotations and get a list of matching documents back. The example domain is software engineering where over 10,000 different artifacts (software code, documentation, user manuals, papers, etc.) were annotated semantically with respect to a domain ontology.  *Figure 3.11* shows a query where the user needs more information about the parameters of a particular component, called Sentence Splitter. The results are a list of document URLs that mention the query concepts. QuestIO implicitly interprets the query as a search for all documents discussing Sentence Splitter parameters.

QuestIO interprets the queries as follows. First it tries to match some or all of the contained words to ontology concepts. Then any remaining textual segments are used to predict property names and act as context for disambiguation. The sequence of concepts and property names can then be converted into a formal query that is executed against the semantically annotated documents. Throughout the process, metrics are used to score the possible query interpretations, allowing the filtering of low scoring options, thus reducing ambiguity and limiting the search space.

Another similar system is SemSearch [44], which is based on Sesame for indexing the semantics and Lucene for indexing the texts. Queries can be a combination of keywords (e.g., news) and connectors such as “and” and “or” (see  *Fig. 3.12* for an example). The system performs semantic matching between words in complex queries and semantic entities by exploring different plausible combinations between the keywords. For longer queries this could compromise the performance of the search engine and more efficient strategies are needed.

In general, natural language interfaces to ontologies, while potentially useful for naive users, need to be evaluated in practice with large number of users, different ontologies, and large document collections, in order to demonstrate clearly their benefits over the other kinds of retrieval interfaces discussed here.

Ontology-Based Faceted Browsing: KIM has a number of front end user interfaces for annotation retrieval and ones customized for specific applications can be easily added.

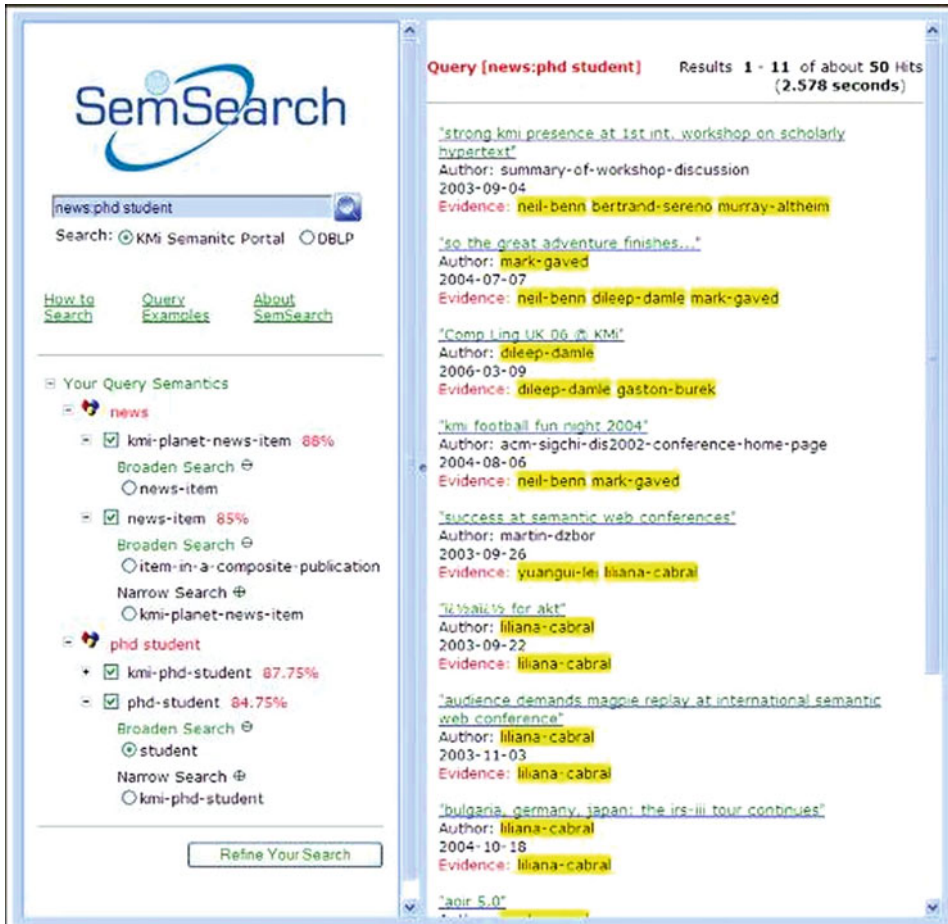
Search knowledge about GATE

sentence splitter parameter Search

Result:	
http://gate.ac.uk/gate/doc/javadoc/gate/creole/class-use/ResourceInstantiationException.html	Source Documentation
http://gate.ac.uk/gate/doc/javadoc/gate/creole/class-use/ExecutionException.html	Source Documentation
http://gate.ac.uk/sale/tao/split.html	Web Page
http://gate.ac.uk/sale/tao/splitch2.html	Web Page

 **Fig. 3.11**

Language-based interface for semantic annotation retrieval



■ Fig. 3.12
Example SemSearch query results

The KIM plug-in for Internet Explorer (see Fig. 3.13) provides lightweight delivery of semantic annotations to the end user. On its first tab, the plug-in displays the ontology and each class has a color used for highlighting the metadata of this type. Classes of interest are selected by the user via check boxes. The user requests the semantic annotation of the currently viewed page by pressing the Annotate button. The KIM server returns the automatically created metadata with its class and instance identifiers. The results are highlighted in the browser window, and are hyperlinked to the KIM Explorer, which displays further information from the ontology about a given instance (see top right window).

The text boxes on the bottom right in Fig. 3.13 that contain the type and unique identifier are seen as tooltips when the cursor is positioned over a semantically annotated entity.

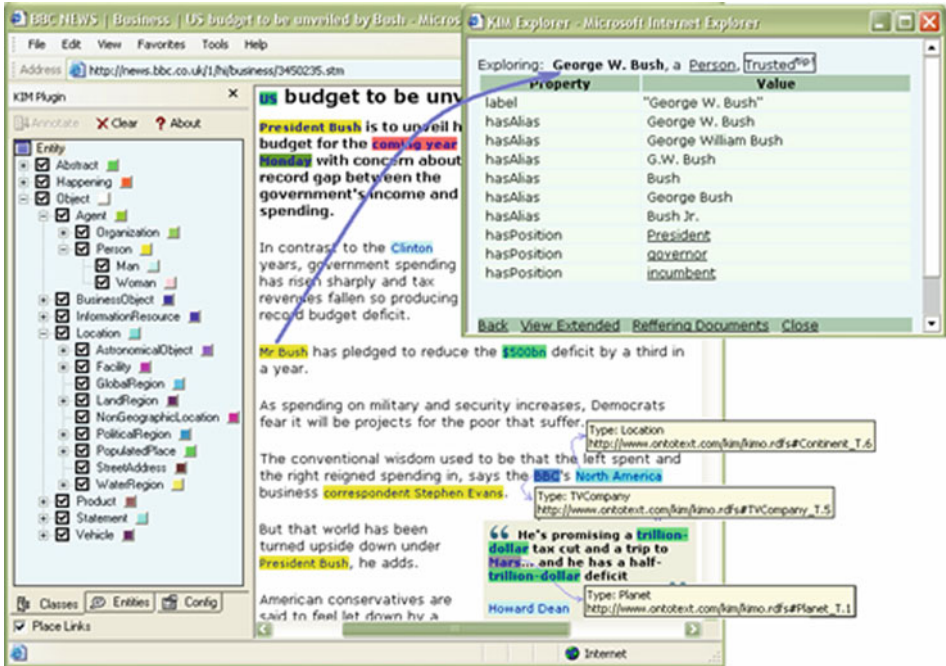


Fig. 3.13

KIM plug-in showing the KIM ontology and KB explorer

KIM also has a comprehensive Web browser-based UI for semantic search. An important part of that is an annotation retrieval interface, similar to faceted search, where the user can select one or more instances (visualized with their RDF labels, but found via their URIs) and obtain the documents where these are all mentioned.

Figure 3.14, for example, shows a case where the user is searching for documents mentioning the entities British Petroleum (BP) and Obama, as well as the keyword spill. As new entities are selected as constraints, the number of matching documents is updated dynamically. At the bottom of the figure, one can see the titles of the retrieved documents and some relevant content from them. The titles can be clicked on in order to view the full document content and the semantic annotations within it. The content of the entity columns (People, Organizations, Locations) is also updated to show only entities contained in the currently retrieved set of documents.

Mimir (see <http://gate.ac.uk/family/>) is a multi-paradigm information management index and repository that can be used to index and search over text, annotations, semantic schemas (ontologies), and semantic metadata (instance data). It allows queries that arbitrarily mix full-text, structural, linguistic, and semantic queries (see Figure 3.20) and that which can scale to gigabytes of text. Its scalability has been tested on semantically annotated data exceeding 10 million documents.

The screenshot shows a web-based search interface for KIM's entity-based, faceted annotation retrieval UI. The search term is "spill". The results are faceted into three categories: People (24 shown below), Organizations (25 of 26 shown below), and Locations (25 of 37 shown below). A "Related Concept" sidebar is also visible on the right. Below the facets, a "Document Keyword Filter" shows the search term "spill". A table of matching documents is displayed, showing 1-3 of 3 documents matching the search criteria.

Date	Title
20-06-2010	BP chief's weekend sailing trip stokes anger at oil company ... hours after a hostile interrogation by a US congressional committee on the Gulf Coast oil spill, have provoked sharp criticism on both sides Atlantic. President Barack Obama's chief...
09-06-2010	Gulf oil spill: Barack Obama and David Cameron move to end rift over BP ... Barack Obama moved to defuse a growing political row over the Gulf of Mexico oil spill yesterday by assuring the prime minister that he is trying to blame Britain...
09-06-2010	Tide of anger may turn an ecological tragedy into a political nightmare

Fig. 3.14

KIM's entity-based, faceted annotation retrieval UI

Since typical semantic annotation projects deal with large quantities of data of different kinds, Mimir provides a framework for implementing indexing and search functionality across all these datatypes, listed below in the order of increasing information density:

Text: All semantically annotated documents have a textual content, and consequently, support for full-text search is required in most (if not all) retrieval use cases. Even when semantic annotations are used to abstract away from the actual textual data, the original content still needs to be accessible so that it can be used to provide textual query fragments in the case of more complex conceptual queries.

Mimir uses inverted indexes for indexing the document content (including additional linguistic information, such as Part-Of-Speech or morphological roots), and for associating instances of annotations with the position in the input text where they occur. The inverted index implementation used by Mimir is based on MG4J (<http://mg4j.dsi.unimi.it/>).

Semantic Annotations: The semantic annotation index supports a more generic retrieval paradigm. A unique feature of Mimir is that it can index linguistic, as well as semantic annotations, which thus enables queries mixing the two. For example, if all words in the indexed documents are annotated according to their part of speech and also with semantic classes such as Person, Location, Organization, then the user can pose Mimir queries such as "CEO of {Mention class==Organization} {Verb}," which mix text (i.e., CEO of) with semantic and linguistic annotations. A unique and important feature of Mimir is the support for queries nesting one annotation within another, for example, retrieving all semantic annotations of type Person that are contained in document titles. Like many other retrieval systems, Mimir also supports Klene operators.

Knowledge Base Data: Knowledge Base (KB) Data consists of an ontology populated with instances. KB data is used to reach a higher level of abstraction over the information in the documents that enables conceptual queries such as finding date ranges or distances. A KB is required for answering such queries because this involves actions like converting from one date format into another and reasoning about scalar values.

A KB that is pre-populated with appropriate world knowledge can perform other generalizations that are natural to human users, such as being able to identify Vienna as a valid answer to queries relating to Austria or Europe.

Mimir uses a Knowledge Base to store some of the information relating to semantic annotations. The links between annotations, the textual data, and the knowledge base information are created by the inclusion into the text indexes of a set of specially created URIs that are associated with annotation data. Furthermore, the URI of entities from the Knowledge Base can be stored as annotation features. The knowledge store used by Mimir for retrieval is based on OWLIM (<http://www.ontotext.com/owlim/>).

Mimir is discussed and exemplified further in [Sect. 3.2.3](#) below.

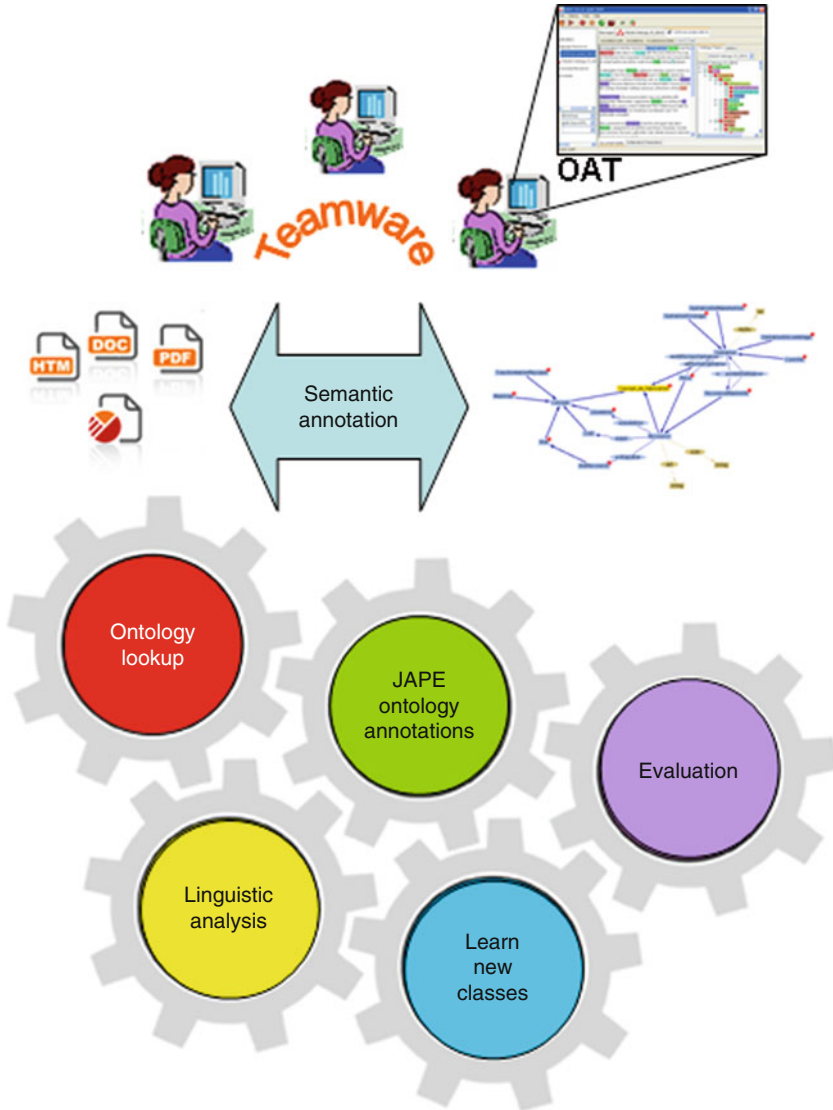
3.2 Example Applications

3.2.1 GATE: A Semantic Annotation Framework

GATE (<http://gate.ac.uk>, [11]) differs from other manual and automatic semantic annotation systems in that it is a framework. In other words, it provides reusable implementations of semantic annotation components and a set of prefabricated software building blocks that researchers can use, extend, and customize for their specific needs. [Figure 3.15](#) shows the main semantic annotation components, which will be discussed in more detail next. Conceptually, one can distinguish tools for: (1) manual semantic annotation (i.e., Teamware and OAT in the top half of the figure); (2) document and ontology editing and visualization (the center of the figure); and (3) algorithms for ontology-based information extraction and evaluation (see the lower half).

GATE is implemented in Java and runs on a wide range of platforms. Another distinguishing characteristic of GATE is its *development environment* (called GATE Developer) that helps users minimize the time they spend building new semantic annotation systems or modifying existing ones, by aiding overall development and providing a debugging mechanism for new modules. Because GATE has a plug-in-based model, this allows for the easy coupling and decoupling of the processors, thereby facilitating comparison of alternative configurations of the system or different implementations of the same module (e.g., different parsers). The availability of tools for the easy visualization of data at each point during the development process aids the immediate interpretation of the results.

GATE is engineered to a high standard and supports efficient and robust semantic annotation. It is tested extensively, including regression testing, and frequent performance optimization. GATE has proved capable of processing gigabytes of text and millions of



■ **Fig. 3.15**
GATE semantic annotation components

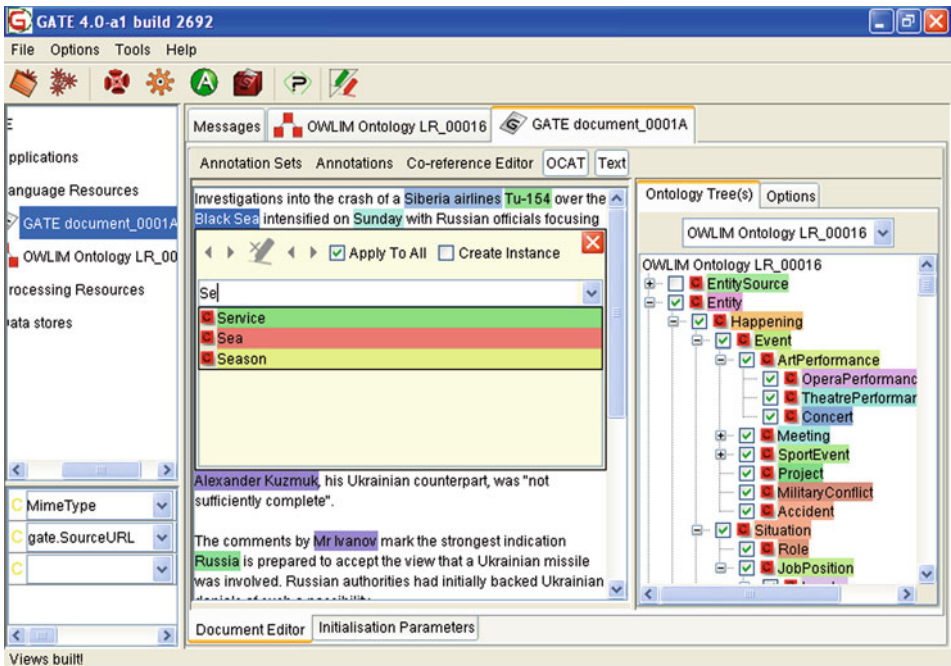
documents. It has been used successfully to build many semantic annotation systems (many discussed in this entry) and ontology learning tools (e.g., Text2Onto [45], Sabou's work [46], SPRAT [47]). The rest of this section discusses the reusable manual annotation tools, the semantic annotation components, and the quantitative evaluation facilities.

First, GATE supports importing, accessing, and visualizing RDF and OWL (see [► KR and Reasoning on the Semantic Web: OWL](#)) ontologies, as well as using those as lexical and reasoning resources within semantic annotation systems. Since the emphasis is

on document annotation, rather than ontology authoring, only basic ontology editing capabilities are provided and the assumption is that typically the ontology would already have been created externally, provided by for example, linked data ontologies. Nevertheless, application-specific extensions with new classes and instances are possible from within GATE.

The second reusable building block is Teamware, which is a collaborative, Web-based annotation tool. It can be used to build both manual and semiautomatic annotation workflows. Alternatively, GATE also provides the single-user, desktop-based Ontology-based Annotation Tool (OAT). [▶ Figure 3.16](#) shows the process of adding a new annotation, that is, the user highlights part of the text and then starts typing the name of the desired class. A list of possible matches is shown for quick selection. It is possible to create new annotations for all occurrences of the selected text within the given document, thus reducing the manual effort. Similarly, the target instance can be specified or a new instance created, if not already available in the ontology. Once an instance is chosen, it is then possible to annotate property values as well. The editor supports manual annotation with respect to more than one ontology at the same time, by adding the ontology URI to each semantic annotation, in addition to the class and instance URIs.

The most reused and extended components for semantic annotation are the automatic ones, especially the ontology-based gazetteers, the JAPE pattern-matching engine, and the machine-learning facilities.



■ Fig. 3.16

OAT: Adding a new annotation

A gazetteer typically contains names of entities/instances such as cities, organizations, days of the week, etc. The word gazetteer is often used interchangeably for both the set of resources that contain the names and for the algorithm that makes use of those lists to find occurrences of these names in documents. GATE's OntoRoot gazetteer analyzes the ontology, that is, all classes, instances, and properties, to derive a list of lexicalizations (e.g., IBM, Big Blue) and their corresponding URIs (in this example, the URI of the IBM instance). In addition, OntoRoot captures morphological variations, for example, the string "language resources" in the document would be matched against a class with label "language resource." A potential limitation of OntoRoot is that it builds the lexical resources from the ontology only once on initialization, which means that any runtime updates to the ontology are not taken into account as soon as they appear.

The JAPE pattern-matching engine uses rules that describe patterns to be matched (left-hand side) and annotations to be created (right-hand side). It provides access to ontologies on the right-hand side of JAPE rules, which allows rules to add new information to the ontology (e.g., add an instance or a newly discovered property value) or to use reasoning (e.g., to obtain semantic distance between concepts). The ontology and most notably the subsumption relation is also taken into account when matching on the left-hand side. So for example, a rule might look for an organization followed by a location, in order to create a `locatedAt` relationship between them. By using subsumption, the rule automatically matches not just organizations, but also all of its subclasses in the ontology, for example, `Company`, `GovernmentOrg`.

The machine-learning components in GATE provide linguistic information as input to a selection of popular machine-learning algorithms directly from GATE's model of annotations. Once collected, the data are exported in the format required by the ML algorithm, which is often a table where each row is an instance and each column is a feature.

When collecting training data, all the annotations of the type specified as instances are found in the given corpus and for each of them the set of attribute values is determined. All attribute values, provided as features to the learning algorithm, refer either to the current annotation or to one situated at a specified relative position (e.g., +1 is the next annotation). The ML implementation has two modes of functioning: training – when the model is being built, and application – when the built model is used to create new annotations. For an example of how learning can be used for semantic annotation see [◆ Sect. 3.1.3](#), as well as [31].

Another key part of the development of semantic annotation systems is quantitative evaluation. The key metrics applied here are precision, recall, and f-measure.

Precision measures the number of correctly identified items as a percentage of the number of items identified. In other words, it measures how many of the items that the system identified were actually correct, regardless of whether it also failed to retrieve correct items. The higher the precision, the better the system is at ensuring that what is identified is correct.

Recall measures the number of correctly identified items as a percentage of the total number of correct items. In other words, it measures how many of the items that should have been identified actually were identified, regardless of how many

spurious identifications were made. The higher the recall rate, the better the system is at not missing correct items.

In general, there is a trade-off between precision and recall, for a system can easily be made to achieve 100% precision by identifying nothing (and so making no mistakes in what it identifies), or 100% recall by identifying everything (and so not missing anything). The F-measure [48] is often used in conjunction with Precision and Recall, as a weighted average of the two.

Since semantic annotation identifies mentions of instances from a given ontology, there are cases when a system would identify an instance successfully but does not assign it the correct class. For example, the entity London in the ontology is an instance of the concept Capital; however, the system annotates the string “London” as belonging to the class City. Since the assigned class does not match the correct class according to the manually annotated data, traditional precision would regard it as wrong. However, due to the closeness of the two classes in the ontology, the system should be given some credit. For such cases, GATE offers BDM (a Balanced Distance Metric), which measures the closeness of two concepts in an ontology or taxonomy [49]. The closer the two concepts are in an ontology, the greater their BDM score is. It is dependent on the length of the shortest path connecting the two classes and also on their depth in the ontology. BDM is normalized with the size of ontology and also takes into account the concept density. In general, BDM can be seen as an improved version of learning accuracy [50].

3.2.2 Large-Scale Semantic Annotation of News

Large-scale semantic annotation produces a lot of metadata in the form of annotations. Processing these does not require a heavy reasoning infrastructure, but a scalable infrastructure for Web crawling, automatic annotation, storage, and retrieval. A particular example application to be discussed here is the annotation of news articles, performed by the KIM platform discussed in ▶ Sect. 3.1.3 above.

The News Collector demonstrator (<http://ln.ontotext.com/>) harvests around a thousand articles daily from the Web and has processed over a million news articles since 2002 (At the time of access, not all these articles were available for retrieval in the online demo.). On average, there were around 30 semantic annotations per document and just over 27 million annotations had to be indexed and stored.

In order to achieve the required scalability and real-time semantic annotation, the system has a cluster architecture, shown in ▶ Fig. 3.17. The cluster provides a set of components that can be configured to work in a distributed environment and allows new processing components to be added on demand. It has centralized repositories for ontologies, semantic annotations, and documents. Scalability of those is achieved through BigOWLIM [51], which is capable of loading and reasoning with over 1 billion of RDF statements. Its performance allows it to replace relational databases in many applications, for example, analytical tasks, business intelligence, and Web front ends to semantic repositories. For an exciting example see BBC’s world cup website, which uses BigOWLIM

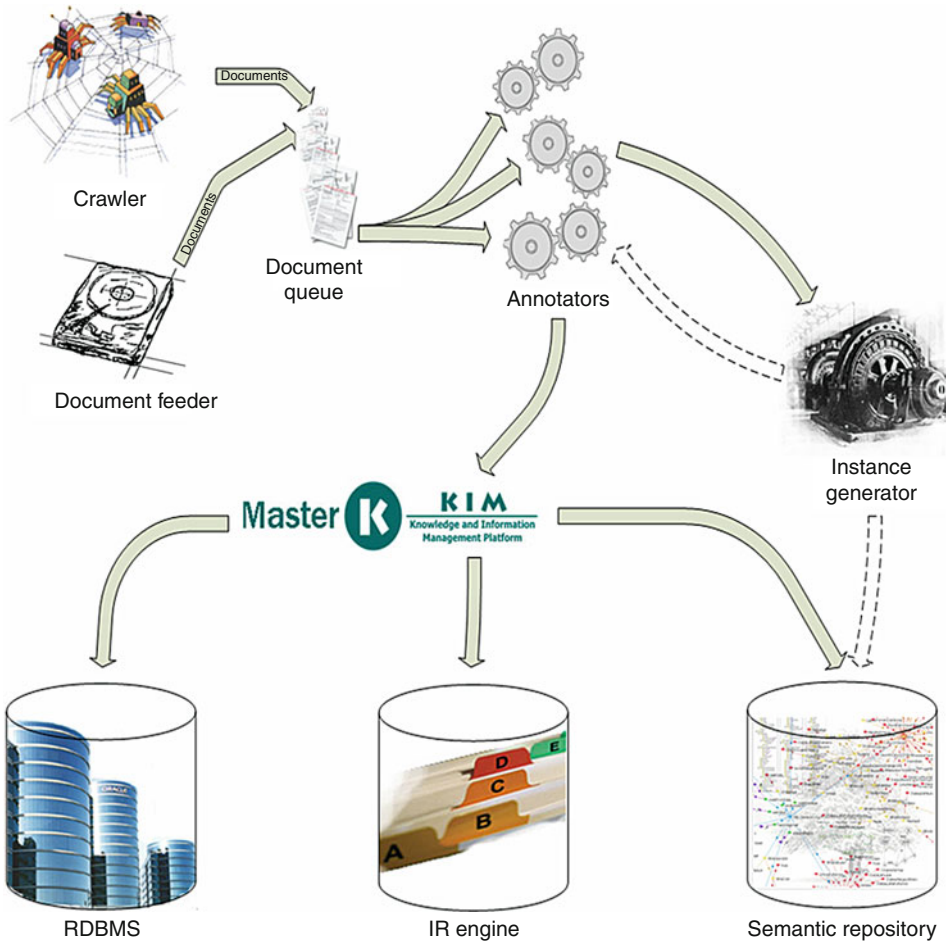


Fig. 3.17
KIM's large-scale cluster architecture

underneath: http://www.bbc.co.uk/blogs/bbcinternet/2010/07/bbc_world_cup_2010_dynamic_sem.html. The semantic metadata are stored in binary files, which allows instant startup and initialization, because it does not require parsing, re-loading and re-inferring all knowledge, unlike triple-based storage formats.

Annotators are the components of the cluster that have an unlimited number of instances, in order to distribute the computationally heavy semantic annotation task. There is also support for multiple Web crawlers and other data feeders. The cluster supports dynamic reconfiguration, that is, the starting of new crawlers and semantic annotators on demand.

As discussed above, the knowledge bases of large-scale applications, such as News Collector, tend to be billions of RDF triples in size. In contrast, “traditional” information extraction methods typically use much smaller-scale lexical resources, especially gazetteer

lists. Therefore, when IE components are adapted to semantic annotation, there is a need for a scalable and fast lookup against the instances in the knowledge base (their labels and properties in particular). Consequently, News Collector has a component called a semantic gazetteer, which runs as one of its annotator modules (others are, e.g., rules for discovery of new instances and relations).

The semantic gazetteer uses the knowledge base to access the entities, their labels, and other properties, as well as some lexical resources (such as possible male person first names). Upon occurrence of a known lexical resource or entity label in the text (e.g., Monday, John, GMT, etc.), the semantic gazetteer generates a semantic annotation with a link to a class in the ontology (e.g., Monday will be linked to the NewsCollector's ontology class `DayOfWeek`). Moreover, where possible, mentions in the text are linked to the specific instances they refer to (e.g., California will be annotated with the URI of the instance `Province.4188`).

Since entities can share labels (e.g., New York is both a state and a city), it is often the case that one named entity reference in the text is associated with several possible types and instances. At this stage all possibilities are generated as separate semantic annotations. A subsequent disambiguation component is applied to filter out the irrelevant annotations, based on the text context and other clues.

In addition, News Collector (and KIM in general) distinguishes between pre-populated (or trusted) instances and instances populated automatically during the semantic annotation process. Since the latter can be much less reliable, they are not used by the semantic gazetteer, in order to reduce the propagation of mistakes.

On the other hand, the discovery of new instances or the enrichment of existing instances with new information extracted from the documents is essential in News Collector and other similar systems that deal with very dynamic domains. Due to the size of their knowledge bases, it is not always practical to carry out ontology enrichment manually. Therefore, discovery of such new information becomes a vital part of the semantic annotation process.

In practical terms, this results in having newly discovered annotations that lack instance information and are thus not linked to the knowledge base via a URI. News Collector uses the IdRF instance disambiguation framework (see [Sect. 3.1.4](#)) to either find a matching existing instance and enrich that with the new information, or to create a new instance in the knowledge base. At the end of the semantic annotation process all annotations are linked to the ontology (via their type/class information) and to the knowledge base (via the instance URI). Any relation annotations discovered in the text are used to enrich the KB with new property values (e.g., to assert that David Cameron is UK's prime minister following the May 2010 elections).

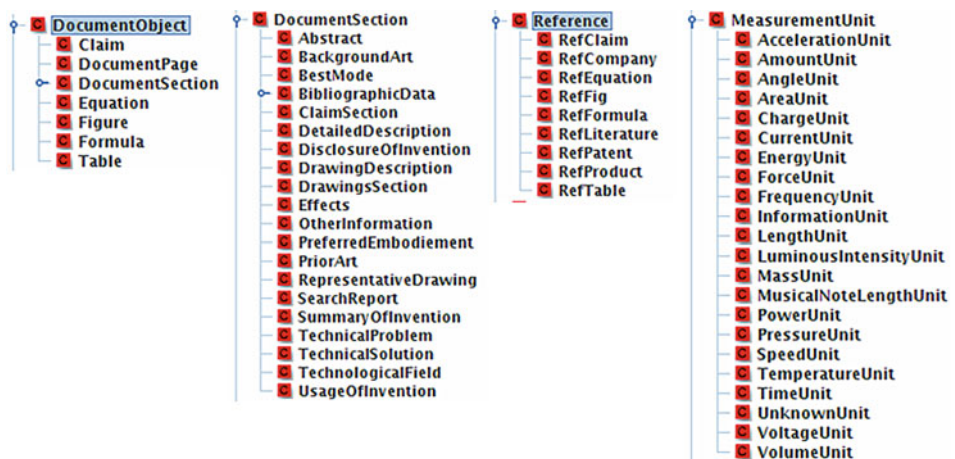
3.2.3 Large-Scale Semantic Patent Processing

Another large-scale application domain is patent processing. The benefits from semantically enriching patents are threefold. First, semantic annotation is capable of dealing with

variable language patterns and format irregularities far easier than text-based regular expressions. For example, references to other patents can be very diverse, for example, US Patent 4,524,128 or Korean laid open utility model application No. 1999-007692. Second, in addition to semantic annotation one can also use an ontology to carry out data normalization. Again, taking an example from references to figures or similarly claims, expressions such as “▶ *Figs. 3.1–3.3*” or “Claims 5–10” imply references not just to the explicitly mentioned figure/claim numbers but also to all those in between. Lastly, automatic semantic annotation techniques are capable of enriching the ever-growing number of patents with more detailed knowledge, which can then be retrieved using multi-paradigm search tools, such as Mimir that combine textual, linguistic, and semantic retrieval.

The SAM project [52] developed an end-to-end, large-scale semantic annotation and retrieval system. One of the main challenges faced in this project is the sheer scale of task. Patent databases typically contain tens of millions of patents, and hundreds of thousands of new ones are produced any year. Worldwide, millions of new patent applications are submitted yearly (see e.g., the statistics page of the World Intellectual Property Organization at <http://www.wipo.int/ipstats/>). Any application aimed at the IP domain requires a good scalability profile if it is to maintain any credibility.

▶ *Figure 3.18* shows the domain ontology, which models key parts of patent documents, that is, sections, claims, references (e.g., to other patents or publications). Measurements are specific to one or more subject areas and are of interest to specialized patent searchers. Currently, patent professionals use traditional keyword search, but face serious difficulties finding measurements reliably, due to the diverse ways in which they are expressed in language and the need for normalization, for example, some patents have metric units, whereas others use imperial measures. Therefore, measurements are an



■ Fig. 3.18

The SAM patent ontology

excellent example of the added value and power of automatic semantic annotation and retrieval methods.

The SAM system was developed using GATE [11] and comprises of three types of components: a tokenizer, gazetteer, and a set of semantic annotation rules. These rules are based on patterns and clue words. For example to locate a reference to a table, one rule looks for the clue word table followed by a number. Gazetteers annotate such clue words in the text with all their inflections.

The reference gazetteers are rather small in size, 314 elements in total, and contain clue words such as *Figure*, *Table*, and *Example* to name a few. They also contain entries such as *described in* or *Patent application no.* to help locate literature and patent references.

In the case of measurements, a database (<http://www.gnu.org/software/units>) containing more than 30 K entries was used to automatically populate a gazetteer list. The database also contains transformation rules for transforming one measurement value into another (e.g., inches to centimeters). Since a gazetteer is simply a list of entries, the information about transforming rules has been populated in the ontology. These rules are used for answering semantic queries by transforming values in one measurement unit into the other on-the-fly.

The application has over 30 rules that identify mentions of the ontology classes in the text (see Fig. 3.19). For example, these include identification of complex equations and intervals of measurements. First, the measurement gazetteer is used for identifying measurement units in the text. In the example below, the pattern would annotate text

The screenshot displays the SAM application interface. At the top, there are four tabs: "Annotation Sets", "Annotations List", "Co-reference Editor", and "Text". Below the tabs is a text area containing a document snippet. The text is annotated with various semantic classes, indicated by colored boxes and arrows pointing to a legend on the right. The legend includes the following items:

- safe.preprocessing (checkbox)
- Measurement (purple box)
- Reference (yellow box)
- Section (green box)

Annotations in the text include:

- "Measurement interval" (purple) pointing to "interval" in "FIG. 4 b".
- "Value" (purple) pointing to "1.2 to 0.5-1 mol%" and "0.25 mol%".
- "Figure reference" (yellow) pointing to "FIG. 4 b".
- "Measurement unit" (green) pointing to "mol%".
- "Patent reference" (yellow) pointing to "U.S. Pat. No. 3,316,279" and "U.S. Pat. Nos. 3,657,292 and 3,646,130".
- "Literature reference" (yellow) pointing to "Chemistry—A European Journal 1996, 2, 168".

Fig. 3.19

A semantically annotated patent

such as “40–50 mph” where 40 and 50 are the two numbers and *mph* is the measurement unit, identified by the gazetteer. As a result, a new annotation of type Measurement will be created, more specifically one of type interval.

```
Rule: MeasurementInterval
(
  {Number}
  {Token.string == "-"}
  {Number}
  {Unit}
):span
-->
:span.Measurement = {type = "interval" }
```

In order to evaluate the consistency in the application’s performance on a large dataset, experiments were carried out on a corpus consisting of 1.3 million US Patent Office documents (108 GB) in XML format with a few attributes on each markup. The average document size was 85 KB. Automatic semantic annotation of all 1.3 million documents took 142 h (5.92 days), at a processing rate of 203.76 KB/s, on a server with 12 threads running in parallel.

In order to be able to estimate the number of semantic annotations that the application produces per document, 20 documents were obtained at random. These contained 147 section annotations, 604 measurements, 1,351 references, and 150,140 linguistic annotations. Based on these results, it would be reasonable to estimate that each document contains an average of 105 semantic annotations and 7,507 linguistic annotations.

The document content, semantic annotations, and linguistic data were indexed with Mimir, in order to enable semantic annotation retrieval. ▶ [Figure 3.20](#), for example, shows a multi-paradigm query consisting of the string “of” followed by a measurement semantic annotation with value within the given interval, which must be contained within the examples section. The results show the matching parts of the documents and the surrounding context. As can be seen, the ontology and reasoning has been used to match values like 2 in. or 135 mm to the query range of between 2.5 and 15 cm.

3.3 Future Issues

The semantic annotation of Web and intranet content is a research problem that has been receiving significant interest over the past 10 years. As discussed in this entry, automatic and manual approaches have often been combined or used independently, depending on the target application, the volume of the target content, and desired accuracy.

Scalability and large volume, real-time semantic annotation were a challenge until relatively recently, but systems, such as NewsCollector and the patent annotator, have now emerged and are capable of dealing with this challenge.

However, with the emergence of the Web of Data, a new challenge has now emerged. While previously the question was how to annotate millions of documents with

MÍMIR



GATE Unified Search

of {Measurement spec="2.5 to 15 cm"} IN {Section type=Examples}

Search

Results 91 - 100 of 3,879

to have an outside diameter of 80mm and a thickness of 10	EP-1829844-A1
5 μm, a size of 34.3 mm x 34.3 mm	EP-1825980-A2
conducted at a span distance of 135 mm and a speed of 1	EP-1825980-A2
sheets each having a size of 30 mm x 10 mm in planar	EP-1829841-A1
) with an internal diameter of 2 Inch and a length of 1	EP-1829854-A1
) with an internal diameter of 2 Inch and a length of 2	EP-1829854-A1
) with an internal diameter of 2 Inch and a length of 1	EP-1829854-A1
) with an internal diameter of 2 Inch and a length of 2	EP-1829854-A1
) with an internal diameter of 2 Inch and a length of 1	EP-1829854-A1
) with an internal diameter of 2 Inch and a length of 2	EP-1829854-A1

Page < Prev 5 6 7 8 9 10 11 12 13 14 Next >

Skip pages 10> 100>

Fig. 3.20

Mímir's annotation retrieval UI

a small-to-medium-sized ontology, the problem is now how to annotate content with respect to large, interlinked ontologies of billions of RDF triples and millions of instances. Ontologies of this size result in significant ambiguities and thus the onus is now on the instance disambiguation algorithms. However, as discussed in [Sect. 3.1.4](#), further research and especially cross-domain, rigorous evaluations are needed. In addition, with linked data becoming an increasingly important publishing format for analysis results, existing semantic annotation systems need to adapt their conceptual modeling and output mechanisms.

Another considerable open issue is the fact that existing Semantic Web ontologies typically contain very limited linguistic information (i.e., labels), which in turn limits their usefulness as a resource for ontology-based information extraction and semantic annotation. Recent work on linguistically grounded ontologies [53] has recognized this shortcoming and proposed a more expressive model for associating linguistic information to ontology elements. While this is a step in the right direction, nevertheless, further work is still required, especially with respect to building multilingual semantic annotation systems.

Change management and the dynamics of semantic annotations is yet another area that needs to be addressed in future work. The problem arises from the dynamic nature of ontologies (and the world in general). For example, if an ontology is updated with new subclasses or an instance is changed to a class, then the question is what changes need to be made to the semantic annotations and/or the automatic software that created them.

Last but not least, there are some technological challenges, especially the problem of delivering semantic annotation using the Software-as-a-Service model. Unfortunately, content analysis services are currently problematic for both suppliers and customers, for two reasons. First, service creation can have high initial and ongoing infrastructural costs, which are only affordable to very few, large companies as a result. Second, existing semantic annotation services mostly focus on English and a couple of other languages. For example, OpenCalais supports only three languages, is not easily customizable by its users, and involves vendor lock-in. All processed content is also made accessible to the provider (Thomson Reuters in this case), which is not always appropriate due to confidentiality. While there are other proven semantic annotation tools, which are open and easily customizable, they are not yet available as scalable Web Services.

3.4 Cross-References

- ▶ [KR and Reasoning on the Semantic Web: OWL](#)
- ▶ [Multimedia, Broadcasting, and eCulture](#)
- ▶ [Semantic Annotation and Retrieval: RDF](#)
- ▶ [Semantic Annotation and Retrieval: Web of Data](#)

References

1. McDowell, L.K., Cafarella, M.: Ontology-driven information extraction with OntoSyphon. In: Proceedings of the Fifth International Semantic Web Conference (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 428–444. Springer, Berlin (2006)
2. Mahesh, K., Kud, J., Dixon, P.: Oracle at TREC8: a lexical approach. In: Proceedings of the Eighth Text Retrieval Conference (TREC-8), Gaithersburg (1999)
3. Voorhees, E.: Using WordNet for text retrieval. In: Fellbaum, C. (ed.) WordNet: An Electronic Lexical Database, pp. 285–303. MIT Press, Cambridge (1998)
4. Handschuh, S., Staab, S.: Authoring and annotation of web pages in CREAM. In: Proceedings of the 11th International World Wide Web Conference (WWW 2002), Honolulu (2002)
5. Uren, V., Cimiano, P., Iria, J., Handschuh, S., Vargas-Vera, M., Motta, E., Ciravegna, F.: Semantic annotation for knowledge management: requirements and a survey of the state of the art. *J Web Semant.* 4(1), 14–28 (2006)
6. Schroeter, R., Hunter, J.: Annotating relationships between multiple mixed-media digital objects by extending Annotea. In: Proceedings of the Fourth European Semantic Web Conference (ESWC 2007), Innsbruck. Lecture Notes in Computer Science, vol. 4519, pp. 533–548. Springer, Berlin (2007)
7. Halaschek-Wiener, C., Golbeck, J., Schain, A., Grove, M., Parsia, B., Hendler, J.A.: Annotation and provenance tracking in semantic web photo libraries. In: Proceedings of the International Provenance and Annotation Workshop (IPAW 2006), Chicago. Lecture Notes in Computer Science, vol. 4145. Springer, Berlin (2006)
8. Defense Advanced Research Projects Agency: Proceedings of the Sixth Message Understanding Conference (MUC-6). Defense Advanced Research Projects Agency, Morgan Kaufmann, California (1995)

9. Marsh, E., Perzanowski, D.: Muc-7 evaluation of IE technology: overview of results. In: Proceedings of the Seventh Message Understanding Conference (MUC-7), Fairfax. http://www.itl.nist.gov/iaui/894.02/related_projects/muc/index.html (1998)
10. ACE: Annotation guidelines for Entity Detection and Tracking (EDT). <http://www ldc.upenn.edu/Projects/ACE/> (2004)
11. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A framework and graphical development environment for robust NLP tools and applications. In: Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL 2002), Philadelphia (2002)
12. Kogut, P., Holmes, W.: AeroDAML: applying information extraction to generate DAML annotations from web pages. In: First International Conference on Knowledge Capture (K-CAP 2001), Workshop on Knowledge Markup and Semantic Annotation, Victoria (2001)
13. Fellbaum, C. (ed.): WordNet – An Electronic Lexical Database. MIT Press, Cambridge (1998)
14. Ciravegna, F., Wilks, Y.: Designing adaptive information extraction for the semantic web in Amilcare. In: Handschuh, S., Staab, S. (eds.) Annotation for the Semantic Web. IOS Press, Amsterdam (2003)
15. Maynard, D., Tablan, V., Cunningham, H., Ursu, C., Saggion, H., Bontcheva, K., Wilks, Y.: Architectural elements of language engineering robustness. *J. Nat. Lang. Eng.* **8**(2/3), 257–274 (2002). Special Issue on Robust Methods in Analysis of Natural Language Data
16. Ciravegna, F.: Adaptive information extraction from text by rule induction and generalisation. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001), Seattle (2001)
17. Ciravegna, F., Dingli, A., Petrelli, D., Wilks, Y.: User-system cooperation in document annotation based on information extraction. In: Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2002), Siguenza, pp. 122–137 (2002)
18. Motta, E., Vargas-Vera, M., Domingue, J., Lanzoni, M., Stutt, A., Ciravegna, F.: MnM: Ontology driven semi-automatic and automatic support for semantic markup. In: Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2002), Siguenza, pp. 379–391 (2002)
19. Handschuh, S., Staab, S., Ciravegna, F.: S-CREAM—semi-automatic CREATION of metadata. In: Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2002), Siguenza, pp. 358–372 (2002)
20. Handschuh, S., Staab, S., Maedche, A.: CREAM – creating relational metadata with a component-based, ontology-driven framework. In: Proceedings of the First International Conference on Knowledge Capture (K-CAP 2001), Victoria (2001)
21. Baumgartner, R., Froelich, O., Gottlob, G.: The Lixto systems applications in business intelligence and semantic web. In: Proceedings of the Fourth European Semantic Web Conference (ESWC 2007), Innsbruck. Lecture Notes in Computer Science, vol. 4519, pp. 16–26. Springer, Berlin (2007)
22. Domingue, J., Dzbor, M., Motta, E.: Magpie: Supporting browsing and navigation on the semantic web. In: Nunes, N., Rich, C. (eds.) Proceedings of the ACM Conference on Intelligent User Interfaces (IUI 2004), Portugal, pp. 191–197 (2004)
23. Gridinoc, L., Sabou, M., D’Aquin, M., Dzbor, M., Motta, E.: Semantic browsing with PowerMagpie. In: Proceedings of the Fifth European Semantic Web Conference on the Semantic Web (ESWC 2008), Tenerife. Lecture Notes in Computer Science, vol. 5021, pp. 802–806. Springer, Heidelberg (2008)
24. Cimiano, P., Handschuh, S., Staab, S.: Towards the self-annotating web. In: Proceedings of the 13th International World Wide Web Conference (WWW 2004), New York (2004)
25. Shchekotykhin, K.M., Jannach, D., Friedrich, G., Kozeruk, O.: AllRight: automatic ontology instantiation from tabular web documents. In: Proceedings of the Sixth International Semantic Web Conference and Second Asian Semantic Web Conference (ISWC/ASWC 2007), Busan. Lecture Notes in Computer Science, vol. 4825, pp. 466–479. Springer, Berlin (2007)
26. Dill, S., Eiron, N., Gibson, D., Gruhl, D., Guha, R., Jhingran, A., Kanungo, T., Rajagopalan, S., Tomkins, A., Tomlin, J.A., Zien, J.Y.: SemTag

- and seeker: boot-strapping the semantic web via automated semantic annotation. In: Proceedings of the 12th International World Wide Web Conference (WWW 2003), Budapest (2003)
27. Mahesh, K., Nirenburg, S., Cowie, J., Farwell, D.: An assessment of Cyc for natural language processing. Technical report MCCS report, New Mexico State University (1966)
 28. Popov, B., Kiryakov, A., Kirilov, A., Manov, D., Ognyanoff, D., Goranov, M.: KIM – semantic annotation platform. In: Proceedings of the Second International Semantic Web Conference (ISWC 2003), Sanibel Island. Lecture Notes in Computer Science, vol. 2870, pp. 484–499. Springer, Heidelberg (2003)
 29. Kiryakov, A., Popov, B., Ognyanoff, D., Manov, D., Kirilov, A., Goranov, M.: Semantic annotation, indexing and retrieval. *J. Web Semant.* 1(2), 671–680 (2004). ISWC 2003 Special Issue
 30. Mikheev, A., Moens, M., Grover, C.: Named entity recognition without gazetteers. In: Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics (EACL 1999), Bergen, pp. 1–8 (1999)
 31. Li, Y., Bontcheva, K., Cunningham, H.: Hierarchical, perceptron-like learning for ontology based information extraction. In: Proceedings of the 16th International World Wide Web Conference (WWW 2007), Banff, pp. 777–786 (2007)
 32. Gruhl, D., Nagarajan, M., Pieper, J., Robson, C., Sheth, A.: Context and domain knowledge enhanced entity spotting in informal text. In: Proceedings of the Eighth International Semantic Web Conference (ISWC 2009), Chantilly. Lecture Notes in Computer Science, vol. 5823, pp. 260–276. Springer, Berlin (2009)
 33. Aswani, N., Bontcheva, K., Cunningham, H.: Mining information for instance unification. In: Proceedings of the Fifth International Semantic Web Conference (ISWC 2006), Athens. Lecture Notes in Computer Science, vol. 4273, pp. 329–342. Springer, Berlin (2006)
 34. Fernandez, N., Blazquez, J.M., Sanchez, L., Bernardi, A.: Identityrank: named entity disambiguation in the context of the NEWS project. In: Proceedings of the Fourth European Semantic Web Conference (ESWC 2007), Innsbruck. Lecture Notes in Computer Science, vol. 4519, pp. 604–654. Springer, Heidelberg (2007)
 35. Yankova, M., Saggion, H., Cunningham, H.: Adopting ontologies for multisource identity resolution. In: Duke, A., Hepp, M., Bontcheva, K., Vilain, M.B. (eds.) Proceedings of the First International Workshop on Ontology-supported Business Intelligence (OBI 2008), Karlsruhe. ACM International Conference Proceeding Series, vol. 308, p. 6. ACM, New York (2008)
 36. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V.C., Sachs, J.: Swoogle: a search and metadata engine for the semantic web. In: Proceedings of the 13th ACM Conference on Information and Knowledge Management (CIKM 2004), Washington, DC (2004)
 37. Hildebrand, M., van Ossenbruggen, J., Hardman, J.: /facet: a browser for heterogeneous semantic web repositories. In: Proceedings of the Fifth International Semantic Web Conference (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 272–285. Springer, Berlin (2006)
 38. Damjanovic, D., Agatonovic, M., Cunningham, H.: Natural language interfaces to ontologies: combining syntactic analysis and ontology-based lookup through the user interaction. In: Proceedings of the Seventh Extended Semantic Web Conference (ESWC 2010), Heraklion. Lecture Notes in Computer Science, vol. 6088, pp. 106–120. Springer, Heidelberg (2010)
 39. Lopez, V., Uren, V., Motta, E., Pasin, M.: Aqualog: an ontology-driven question answering system for organizational semantic intranets. *J. Web Semant.* 5(2), 72–105 (2007)
 40. Kaufmann, E., Bernstein, A.: How useful are natural language interfaces to the semantic web for casual end-users? In: Proceedings of the Fourth European Semantic Web Conference (ESWC 2007), Innsbruck. Lecture Notes in Computer Science, vol. 4519. Springer, Berlin (2007)
 41. Funk, A., Tablan, V., Bontcheva, K., Cunningham, H., Davis, B., Handschuh, S.: CLONe: controlled language for ontology editing. In: Proceedings of the Sixth International Semantic Web Conference (ISWC 2007), Busan. Lecture Notes in Computer Science, vol. 4825, pp. 142–155. Springer, Berlin (2007)
 42. Bernstein, A., Kaufmann, E.: GINO – a guided input natural language ontology editor. In: Proceedings of the Fifth International Semantic Web

- Conference (ISWC 2006), Athens. Lecture Notes in Computer Science, vol. 4273, pp. 144–157. Springer, Berlin (2006)
43. Damljanovic, D., Bontcheva, K.: Enhanced semantic access to software artefacts. In: Proceedings of the Fourth International Workshop on Semantic Web Enabled Software Engineering (SWESE 2008), Karlsruhe (2008)
 44. Lei, Y., Uren, V., Motta, E.: Semsearch: a search engine for the semantic web. In: Managing Knowledge in a World of Networks, pp. 238–245. Springer, Berlin/Heidelberg (2006)
 45. Cimiano, P., Voelker, J.: Text2Onto – a framework for ontology learning and data-driven change discovery. In: Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB 2005), Alicante (2005)
 46. Sabou, M.: From software APIs to web service ontologies: a semi-automatic extraction method. In: Proceedings of the Third International Semantic Web Conference (ISWC 2004), Hiroshima. Lecture Notes in Computer Science, vol. 3298, pp. 410–424. Springer, Berlin (2004)
 47. Maynard, D., Funk, A., Peters, W.: Using lexico-syntactic ontology design patterns for ontology creation and population. In: Proceedings of the ISWC Workshop on Ontology Patterns (WOP 2009), Washington, DC (2009)
 48. van Rijsbergen, C.: Information Retrieval. Butterworths, London (1979)
 49. Maynard, D., Peters, W., Li, Y.: Metrics for evaluation of ontology-based information extraction. In: Proceedings of the Fourth International Workshop on Evaluation of Ontologies for the Web (EON 2006) at the 15th International World Wide Web Conference (WWW 2006), Edinburgh (2006)
 50. Cimiano, P., Staab, S., Tane, J.: Automatic acquisition of taxonomies from text: FCA meets NLP. In: Proceedings of the ECML/PKDD Workshop on Adaptive Text Extraction and Mining, Cavtat-Dubrovnik, Croatia, pp. 10–17 (2003)
 51. Kiryakov, A.: OWLIM: balancing between scalable repository and light-weight reasoner. In: Proceedings of the 15th International World Wide Web Conference (WWW 2006), Edinburgh (2006)
 52. Agatonovic, M., Aswani, N., Bontcheva, K., Cunningham, H., Heitz, T., Li, Y., Roberts, I., Tablan, V.: Large-scale, parallel automatic patent annotation. In: Proceedings of First International CIKM Workshop on Patent Information Retrieval (PaIR 2008), Napa Valley (2008)
 53. Buitelaar, P., Cimiano, P., Haase, P., Sintek, M.: Towards linguistically grounded ontologies. In: Proceedings of the Sixth European Semantic Web Conference (ESWC 2009), Heraklion. Lecture Notes in Computer Science, vol. 5554, pp. 111–125. Springer, Heidelberg (2009)

4 Semantic Annotation and Retrieval: RDF

Fabien L. Gandon¹ · Reto Krummenacher² · Sung-Kook Han² ·
Ioan Toma²

¹INRIA-Edelweiss, Sophia Antipolis, France

²Universität Innsbruck, Innsbruck, Austria

4.1	<i>Introduction</i>	119
4.2	<i>A Technical Overview of RDF</i>	121
4.2.1	A Model for a Resource Description Framework	121
4.2.1.1	Triples as Atoms of Knowledge	121
4.2.1.2	A Graph-Oriented Data Model	123
4.2.1.3	Identifying Conceptual Vocabularies Through Namespaces	124
4.2.1.4	Typing and Multi-instantiation	125
4.2.1.5	Existential Quantification of Resources: Blank Nodes	125
4.2.2	Serializing RDF Graphs	126
4.2.2.1	RDF Graphs as XML Trees	126
4.2.2.2	Triple Serialization in N-Triples	127
4.2.2.3	Triple Serialization in Turtle	128
4.2.3	Complex Data Structures	129
4.2.3.1	Open Containers	129
4.2.3.2	Closed Collections	130
4.2.3.3	Semantic-Less Reification	130
4.2.4	Lightweight Ontology Formalization with RDFS	132
4.2.4.1	Taxonomical Skeleton of Resource Classes	132
4.2.4.2	Taxonomical Skeleton of Resource Relations	133
4.2.5	Limitations of RDF/S in Building a Logical Framework	136
4.3	<i>Examples and Applications</i>	138
4.3.1	Data Integration on the Web	138
4.3.1.1	Schema and Data Samples	139
4.3.2	RDF Web Application	146
4.4	<i>Related Resources</i>	149
4.4.1	Books	149
4.4.2	Softwares and Tools	150

4.4.3	Websites	151
4.4.4	Standards	152
4.5	<i>Future Issues</i>	152
4.6	<i>Summary</i>	154
4.7	<i>Cross-References</i>	154

Abstract: The Resource Description Framework (RDF) is the de facto standard for metadata on the Web. Both, RDF and its schema language RDFS are recommended by W3C for interlinking resources on the Web and for fostering interoperability among distributed data sources. For this purpose, RDF relies on URIs for identifying resources, and constitutes a graph-based data model for linking such resources. To this end, RDF provides the fundamental building blocks for the graph-based data structures that are leveraged by the Semantic Web. More recently, RDF is no longer only the base layer of the Semantic Web, but its importance has increased, and nowadays RDF provides the principal data model for almost all data-minded protocols and formats that are promoted and standardized by W3C.

4.1 Introduction

The Resource Description Framework (RDF) is a framework to publish statements on the Web about anything. It allows anyone to describe resources, in particular Web resources, such as the author, creation date, subject, and copyright of an image. RDF was first published by W3C in 1999 shortly after the first recommendation of the XML Syntax in 1998. The RDF recommendation emerged from work such as the Channel Definition Format (CDF, www.w3.org/TR/NOTE-CDFsubmit.html) and the Meta Content Framework (MCF, <http://www.w3.org/TR/NOTE-MCF-XML-970624/>). MCF was first introduced by Apple Computer in 1996, as metadata system for the Web that was also adapted by Netscape even before submitting it to W3C. Insights from the Dublin Core community and from PICS (www.w3.org/PICS), the Platform for Internet Content Selection, were also key in shaping the direction of the RDF project. The PICS specification enabled labels (metadata) to be associated with Internet content and was primarily defined for access control to content. PICS has since been superseded by the Protocol for Web Description Resources (POWDER, www.w3.org/2007/powder/) that was chartered in 2007. The principal idea of RDF was to define a new framework for viewing, manipulating, and associating networked collections of distributed information. As such, it provides interoperability between applications that exchange machine-understandable information. This vision was initially described in the Semantic Web Road map of Tim Berners-Lee (<http://www.w3.org/DesignIssues/Semantic.html>). From these early days, RDF has evolved to the 2004 recommendation that is nowadays considered to be the RDF standard, still responding to the same objectives of interlinking of information and the provisioning of interoperability. Effectively, the initial specification is only known to few insiders. Although stable for many years now, RDF is still evolving, and in 2010 W3C called for a workshop on discussing future directions and needs of the Resource Description Framework.

Any information portal or data-based website can be interested in using the graph model of RDF to open its silos of data about persons, documents, events, products, services, places, etc. RDF reuses the Web approach to identify resources (URI) and to allow one to explicitly represent any relationship between two resources. Such statements can come from any source on the Web and be merged with other statements supporting worldwide data integration. By using and reusing URIs, anyone can say anything about

any topic, anyone can add to it, and so on. Additionally, using RDFS, one can define domain-specific classes and properties to describe these resources and organize them in hierarchies. These schemas are also published and exchanged in RDF. RDF not only provides a graph model to publish and link data on the Web, it also provides the fundamental shared data model on which other capabilities are built: querying (SPARQL is built on top of RDF), embedding (RDFa and GRDDL rely on the RDF model), and reasoning (RDFS and OWL are defined on top of RDF). Semantic Web is a Web to link data and share the semantics of their schemas. RDF provides a recommendation to publish and link data. RDFS provides a recommendation to share the semantics of their schemas. The couple RDF and RDFS is also reused in several other activities of W3C, for example, in the Semantic Web Activities, as de facto standard for metadata, and as provider of the fundamental layer for producing machine-interpretable information; hence initiatives around RDFa, OWL, or RIF are very closely related to RDF. Note that the above are covered in the following chapters: SPARQL: [Querying the Semantic Web: SPARQL](#); RDFa and GRDDL [Semantic Annotation and Retrieval: Web of Hypertext – RDFa and Microformats](#); OWL [KR and Reasoning on the Semantic Web: OWL](#); and RIF [KR and Reasoning on the Semantic Web: RIF](#). The Media Annotation working group develops ontologies for cross-community data integration of information related to media objects in the Web, such as video, audio, and images. The Protocols and Formats working group works on universal access and interoperability across the Web, and promotes RDF for the specification of roles and access control. The privacy preference language of P3P uses RDF. Although RDF was initially published as a base layer of the

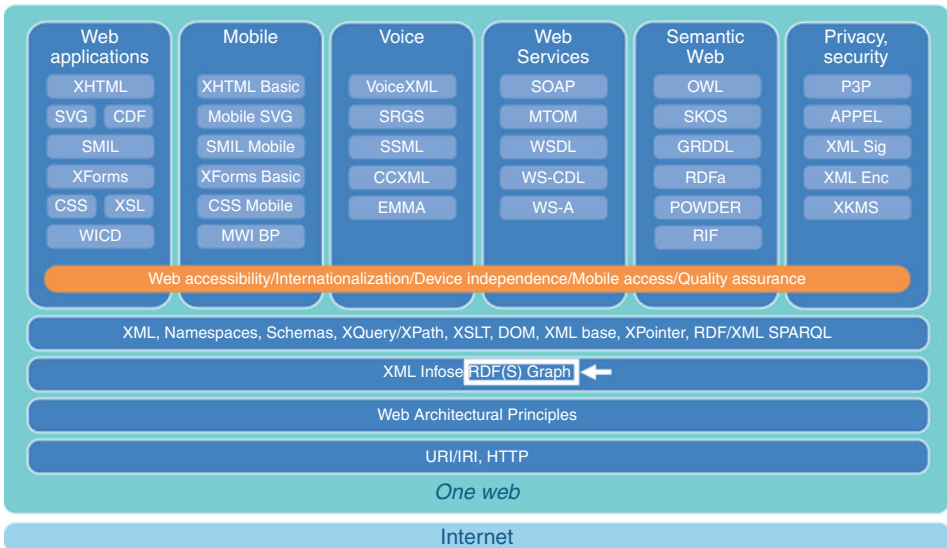


Fig. 4.1

RDF graphs at the foundations of the recommendation stack at W3C © (www.w3.org/2004/10/RecsFigure.png)

Semantic Web standards, it evolved to a general-purpose graph data model that is leveraged as one of two ways for structuring data (XML infosets and RDF graphs) in all known Web standards (➤ [Fig. 4.1](#)).

Besides being naturally pushed by W3C, RDF becomes increasingly important in various other industry and governmental bodies too, such as the open government data movement, the networked economy, pharmaceutical research and development, energy data exchange, and sensor and aggregator networks. All these large-scale data management scenarios require an interoperable, flexible, extensible, and hence schema-last data model – which is precisely what RDF offers.

4.2 A Technical Overview of RDF

4.2.1 A Model for a Resource Description Framework

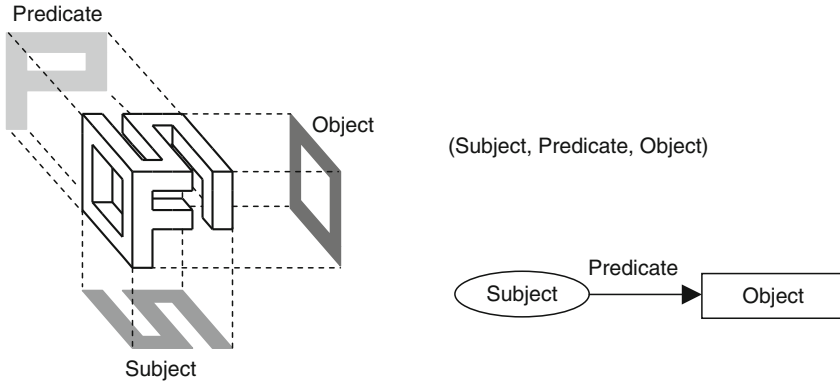
The abbreviation RDF stands for the “Resource Description Framework” in the sense that:

- *Resources* are a core concept on the Semantic Web: everything one might refer to is considered a resource; for example, a Web page, an image, videos, but also a person, a place, a device, an event, an organization, a product, or a service. More technically speaking, everything that can be identified by a URI can be considered a resource.
- *Descriptions* of resources are essential for understanding and reasoning about them. In the most general case, a description is a set of attributes, features, and relations concerning the resource.
- The *Framework* means it provides models, languages, and syntaxes for these descriptions.

In short, RDF provides a standard data structure and a model to encode data and metadata about any subject on the Web. To this end, annotated resources on the Web are then connected to other annotated artifacts, spanning a network of connected resources which links typed information. Discovering such implicit knowledge and making it explicitly available to interested parties via formal representations and Web applications is the main driver of current Semantic Web research. Besides a cloud of facts that are available through the Semantic Web, more and more ontologies are published that allow making sense out of the Web of Data. While ontologies – in the simplest case written as RDF Schemas (RDFS) – are thus very important for interpreting data and for deducing knowledge out of available facts, RDF is the de facto standard for metadata and annotating things on the Web, and hence the lingua franca of open data.

4.2.1.1 Triples as Atoms of Knowledge

RDF graphs are constructed out of so-called RDF triples (➤ [Fig. 4.2](#)). RDF triples describe and connect objects via the combination of resources, properties, and property values; such triples are also referred to as statements. The resource is the subject of the statement,



■ Fig. 4.2

The RDF triple: the atom of knowledge on the Semantic Web inspired by [3]

the property is the predicate of the statement, and the property value is the object of the statement. Therefore, the basic data structure of RDF is a triple of the form `<subject, predicate, object>`.

For example, the assertion “Fabien has written a page `doc.html` about music” can be broken down into two RDF statements about the document: `(doc.html, author, Fabien)` and `(doc.html, theme, music)`. Here again, the resource `doc.html` is the subject, the attribute `theme` is the predicate, and the third element (`music`) is the object of the statement.

Being a Web-oriented framework, RDF identifies resources and properties using URIs, optimally even URLs that allow the dereferencing of the identified Web resources and to discover RDF descriptions – this process is pretty much comparable to the traversal of hyperlinks in HTML Web documents. As stated previously, shared resources across subjects and objects are one of the fundamental principles that allow for constructing RDF graphs. For this reason, URIs are found as the subjects and objects of triples, and in RDF also as unique identifiers for predicates. In RDF, besides being URIs, objects can also take the form of so-called literals, that is, arbitrary typed or untyped strings. Reconsidering the example triple `(doc.html, author, Fabien)`, the subject would be an HTML resource on the Web, while the property would be identified with a globally unique URI. The object of the given triple could, as explained above, be either a resource itself or a string literal to form one of the following triples: `(<doc.html> <author> <Fabien>)` or `(<doc.html> <author> “Fabien”)`. In the former case, `<Fabien>` is a resource and hence a node in the RDF graph that could be dereferenced to discover further facts about the object Fabien; in the latter case, the object is given as a string and becomes a stub in the RDF graph, as the literal “Fabien” does not denote a shared resource.

RDF triples are axiomatic statements, facts that can be seen as binary predicates in logics. An important aspect of the logical view of the RDF triples is that RDF makes an open-world assumption: as opposed to the closed-world assumption of classical systems the absence of a triple is not significant; that is, if a triple does not explicitly claim a fact, it does not mean that the fact could not be true. In other words, the RDF semantics

assumes that whatever is not explicitly stated could be true, while in relational models the assumption is the other way around: facts that are not explicitly claimed are false. In the example of [Table 4.1](#), the fact that one only knows the authors Fabien and York does not mean that there are no other authors; it only means that there are at least the two named authors.

4.2.1.2 A Graph-Oriented Data Model

RDF triples can be seen as two vertices and one arc of a graph describing and linking resources. More technically speaking, RDF is a decentralized data representation model relying on distributed triples that form a global graph. The identity of the resources is based on the URI mechanism: if two resources have the same URI they are one and the same node in the graph. This extremely simple and powerful data representation mechanism has recently found adopters in a large range of domains.

An RDF graph is:

- A multi-graph: a graph that can contain both multiple edges and loops between its vertices
- A directed graph: every edge is oriented going from the end-vertex representing the subject to the end-vertex representing the object
- A labeled graph: edges are labeled with URIs, vertices are labeled with URIs, blank node identifiers, or literals

■ **Table 4.1**

Simple RDF example

English: "The report doc.html has the authors Fabien and York, the theme music and is 23-pages long."	
Logic predicates: Report(doc.html) creator(doc.html,Fabien) creator(doc.html,York) theme(doc.html,Music) nbPages(doc.html, 23)	Triples: (doc.html, type, Report) (doc.html, creator, Fabien) (doc.html, creator, York) (doc.html, theme, Music) (doc.html, nbPages, "23")
<pre> graph LR A([http://.../doc.html]) -- type --> B([http://...#Report]) A -- creator --> C([http://...#fabien]) A -- creator --> D([http://...#york]) A -- theme --> E([http://...#music]) A -- nbPages --> F["23"] </pre>	

From the graphical notation point of view, RDF graphs are directed labeled graphs where resource nodes are depicted as ovals, predicates label the directed arcs depicted as arrows, and literals are stubs and depicted as rectangles (cf. ▶ [Table 4.1](#)).

4.2.1.3 Identifying Conceptual Vocabularies Through Namespaces

As stated previously, URIs are used to identify a diversity of resources. One special case is when a URI is used to identify a set of terms, a vocabulary, a schema; in that case the URI is called a namespace. Namespaces are used in particular to identify the schemas declaring the types of resources and types of relations used to label RDF graphs. In XML documents namespaces are associated to prefixes in order to shorten the resource identifiers in the document, locally using the prefix instead of the full URI. For instance, RDF provides an elementary typing primitive `type` that belongs to the core RDF vocabulary which is identified by the URI <http://www.w3.org/1999/02/22-rdf-syntax-ns#>. Consequently, the `type` predicate can be identified as `rdf:type` instead of <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>. In the continuation of this chapter we use the following:

• <code>rdf</code>	http://www.w3.org/1999/02/22-rdf-syntax-ns#
• <code>rdfs</code>	http://www.w3.org/2000/01/rdf-schema#
• <code>foaf</code>	http://xmlns.com/foaf/0.1/
• <code>dc</code>	http://purl.org/dc/elements/1.1/
• <code>xsd</code>	http://www.w3.org/2001/XMLSchema#
• <code>exs</code>	http://example.org/schema#
• <code>exr</code>	http://example.org#

By using these namespaces and the associated prefixes, the RDF triples of ▶ [Table 4.1](#) can be rewritten more correctly as given in the example of ▶ [Table 4.2](#); all resources are now identified with a URI and denoted by the short form `prefix:localname` of the namespace mechanism that is inherited from XML.

■ **Table 4.2**

Rewritten example using namespaces and URIs

<p>Triples</p> <p>(http://example.org/doc.html , <code>rdf:type</code> , <code>exs:Report</code>)</p> <p>(http://example.org/doc.html , <code>dc:creator</code> , <code>exr:Fabien</code>)</p> <p>(http://example.org/doc.html , <code>dc:creator</code> , <code>exr:York</code>)</p> <p>(http://example.org/doc.html , <code>exs:theme</code> , <code>exr:Music</code>)</p> <p>(http://example.org/doc.html , <code>exs:nbPages</code> , "23")</p>

4.2.1.4 Typing and Multi-instantiation

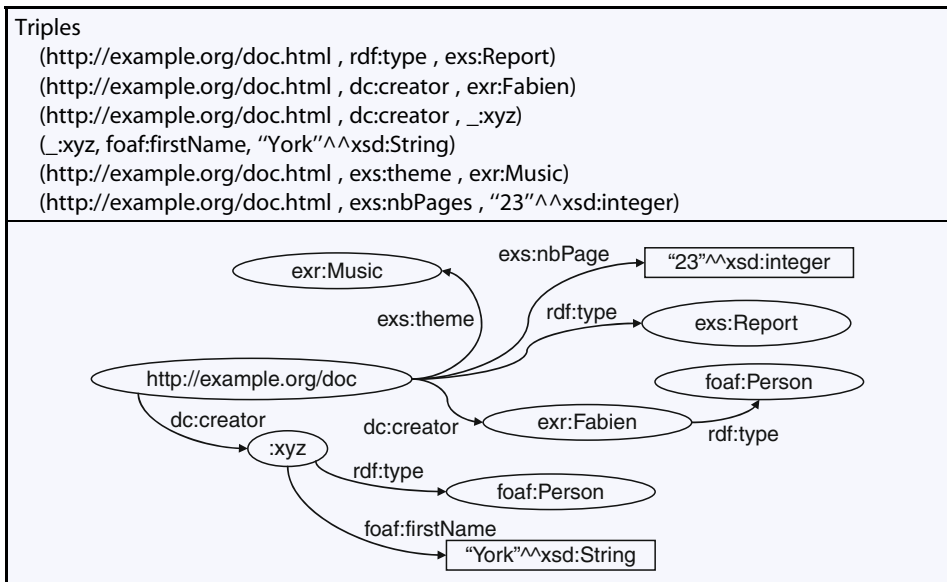
The example of [Table 4.2](#) depicts how the RDF typing primitive `rdf:type` allows for attaching schema information in the form of one or more types to a resource: `http://example.org/doc.html` is declared as being a report: that is, a resource of type `exs:Report`. One important and very powerful difference between typing in RDF and typing in most common object-oriented programming languages is that an RDF resource can belong to several unrelated classes, as RDF allows for multi-instantiation. Note that literals, although not denoting objects, can also be typed. Primitive data values are mostly typed by means of the XML Schema datatypes, as shown in [Table 4.3](#).

4.2.1.5 Existential Quantification of Resources: Blank Nodes

So far in this chapter, all resources were denoted by URIs. In some cases this unified identification of resources is not possible. There are situations when only the type of object, not however a particular instance, is known, or when the instance is anonymous. The RDF response to this is so-called blank nodes. In [Table 4.3](#), the blank node is given as `_:xyz`. The underscore represents the unknown namespace, and hence, the anonymous identifier does not assume a globally unique name. Effectively, blank nodes can only be within a predefined context (e.g., the RDF file in which it was declared), as beyond this

Table 4.3

Introducing typed literals and RDF blank nodes



boundary a blank node loses its point of reference; that is, the vertex it represents. In consequence, blank nodes in different graphs or files, although potentially carrying the same ID, denote two distinct resources, as defined by the RDF semantics. To this end, a blank node is like an existential quantification in logics, meaning, “there exists a resource such that. . .” the indicated typing and properties are fulfilled. In [Table 4.3](#), the author with name York is given by a blank node, and hence the graph states that there exists a resource such that it is a foaf:Person that has the first name “York.”

Although generally quite convenient, blank nodes are discouraged in practice as they break the graph. As stated, blank nodes cannot be reused outside the RDF file in which they were declared, and thus, they prevent the simple extension of graphs and anonymously declared resources. For example, anyone can easily add information about Fabien by expanding the knowledge about the resource `exr:Fabien`; however, no one could add any knowledge about the person York in an open environment, as the blank node is not referenceable beyond the given file. This unfortunately counteracts the idea of extensibility and reusability that are core principles of the very successful Linked Open Data initiative (www.linkeddata.org).

4.2.2 Serializing RDF Graphs

In previous sections, RDF was either represented as triples of the form `<subject, predicate, object>`, or as directed graphs. The graph notation is very useful for representing RDF in particular as it is understandable by humans. However, the RDF graph notation provides an abstract model only that is not machine-interpretable, and hence violates another core principle of RDF. Being understandable by machines also implies being interoperable and exchangeable across distributed and heterogeneous systems and applications, and obviously the Web. For this purpose, there are standardized serialization mechanisms defined, some of them published as W3C recommendations, for converting the abstract RDF graph into a concrete exchange format.

4.2.2.1 RDF Graphs as XML Trees

The only officially standardized serialization of RDF, published by W3C in the RDF/XML Syntax Specification in 2004, is the RDF/XML syntax (www.w3.org/TR/rdf-syntax-grammar/). The principal aim of RDF/XML is to be machine-processable and compliant to the de facto standard of Web document formatting, XML. This allows RDF documents to be easily exchanged between very different types of systems and applications. Note that RDF/XML may be found cumbersome to read but it is not intended for human consumption. Furthermore, criticisms arise as RDF/XML is very verbose, hindering expressivity, as XML forces arbitrary RDF graphs to be represented in a tree. Moreover, there exist several possibilities to express the same RDF graph; that is, the serialization is not unique

Table 4.4

RDF/XML serializations

<pre> <?xml version="1.0"?> <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:exs="http://example.org/schema#"> <rdf:Description rdf:about="http://example.org/doc.html"> <rdf:type rdf:resource="http://example.org/schema#Report"/> <exs:theme rdf:resource="http://example.org#Music"/> <exs:theme rdf:resource="http://example.org#History"/> <exs:nbPages rdf:datatype="http://www.w3.org/2001/XMLSchema#int">23</exs:nbPages> </rdf:Description> </rdf:RDF> </pre>
<pre> <?xml version="1.0"?> <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:exs="http://example.org/schema#"> <exs:Report rdf:about="http://example.org/doc.html" exs:nbPages="23"> <exs:theme rdf:resource="http://example.org#Music"/> <exs:theme rdf:resource="http://example.org#History"/> </exs:Report> </rdf:RDF> </pre>

(cf. [Table 4.4](#)). Several alternative serializations were developed over time; some of them are presented later in this section, although none of them is standardized yet. The reader is also referred to the discussion of future issues around RDF formats in [Sect. 4.5](#).

[Table 4.4](#) shows two possible RDF/XML serializations of the same RDF graph. Every RDF/XML document has a root element `<rdf:RDF>` to declare the XML document to represent RDF. Attributes to the root element are used to specify namespace information and prefixes; recall, RDF inherits the namespace mechanism. A simple description uses the `<rdf:Description>` element with the attribute `rdf:ID` (with a simple ID that would be expanded to a full URI by means of the base namespace of the XML document) or `rdf:about` (with a URI) to identify resources and defines child elements (properties) to describe the resource. The references via `rdf:ID` or `rdf:about` allow the specification of graphs that are hidden in XML trees. Alternatively, and here it starts to get complicated, the `<rdf:Description>` and `<rdf:type>` elements that are used in the first example of [Table 4.4](#) could be merged as a class element for simplicity (e.g., `<exs:Report>`). Moreover, property elements may be declared as attributes of the description or as child elements. An example is given by the `exs:nbPages` property in [Table 4.4](#).

4.2.2.2 Triple Serialization in N-Triples

N-Triples provides a line-based, plain text format for serializing RDF graphs. N-Triples is designed to be a fixed subset of N3 and also subset of Turtle ([Sect. 4.2.2.3](#)) (Notation-3,

The `@prefix` directives allow for the introduction of namespace bindings. Through these declarations, URIs can be abbreviated effectively such as the given `rdf:type` and `exs:Report` resources. Further shortcuts are supported for repeated subjects, or multiple objects. To abbreviate multiple statements with the same subject, Turtle provides a semicolon (;) notation which makes it possible to list predicate–object pairs for the same subject. Multiple statements with the same subject–predicate pair but different objects can be similarly abbreviated using the comma (,) notation; examples for both shortcuts are given above. Further syntactical constructs permit abbreviated notations for blank nodes (`[]`), `rdf:type` (stated as ‘`a`’) or parenthesis for `rdf:Lists` (▶ Sect. 4.2.3.2). The sum of these abbreviations makes Turtle the simplest and most compact RDF serialization format. It effectively reduces the size of the serialized stream. In addition, Turtle is argued to be the most human-readable syntax. For more information about the Turtle syntax, please refer to the authoritative specification site and team submission page by Beckett and Berners-Lee at www.w3.org/TeamSubmission/turtle/.

4.2.3 Complex Data Structures

In order to group and order RDF triples, there are a couple of syntactical primitives defined to provide containers and collections in the form of lists. Moreover, RDF offers a means to make statements about statements via the so-called reification. These special purpose data structures are the subject of this section.

4.2.3.1 Open Containers

Containers are open groups and contain unspecified numbers of resources or literals and possibly duplicates. Although there are three types of containers with distinct meaning in the human sense, RDF does not make any difference in its interpretation. Essentially, the semantics of the three types of containers is identical, and the different classes may be used informally only; that is, for human consumption. For this purpose the following containers are distinguished:

- `rdf:Bag` defines an unordered group of resources or literals.
- `rdf:Seq` defines an ordered group of resources or literals.
- `rdf:Alt` represents a group of resources or literals that are alternatives; that is, only one of the values can be selected.

In any case, a resource is typed as `rdf:Bag`, `rdf:Seq` or `rdf:Alt`, and, syntactically, the members of the container are attached to it via special purpose numbered membership properties of the form `rdf:_1`, `rdf:_2`, etc. (▶ Table 4.5). The RDF/XML syntax provides the property `rdf:li` to avoid the explicit numbering of members. Each occurrence of `rdf:li` is internally turned into a numbered property `rdf:_1`, `rdf:_2`, etc. to form the corresponding RDF graph (▶ Table 4.1).

Table 4.5

Open RDF containers

<pre>Turtle <http://example.org/doc.html> a exs:Report ; dc:creator [a rdf:Bag ; rdf:_1 <http://example.org#Fabien> ; rdf:_2 <http://example.org#York>].</pre>
<pre>RDF/XML <exs:Report rdf:about="http://example.org/doc.html"> <dc:creator> <rdf:Bag> <rdf:li rdf:resource="http://example.org#Fabien"/> <rdf:li rdf:resource="http://example.org#York"/> </rdf:Bag> </dc:creator> </exs:Report></pre>

4.2.3.2 Closed Collections

While containers remain open per definition, as there is no way to restrict the number of members a container contains. Collections, on the contrary, provide closed lists of resources or literals, possibly with duplicates, and contain, as per the RDF recommendation, only the specified members. In RDF, lists are of type `rdf:List` with the property `rdf:first` pointing to the first member and a property `rdf:rest` recursively pointing to the list of the remaining members or to `rdf:nil` if the end of the list is reached; `rdf:nil` represents an empty list. Note that per specification, it is not required that there be only one first element of a list structure, or that a list structure has a first element at all. As mentioned previously, the Turtle syntax offers a parenthesis-based abbreviation for lists. Similarly, RDF/XML offers a convenience notation. Collections are declared as nested elements included via a property that has the attribute `rdf:parseType = "Collection"`. A corresponding example is given in [Table 4.6](#).

4.2.3.3 Semantic-Less Reification

In addition to making statements about Web resources, RDF can be used for making statements about other RDF statements. In order to allow for this, RDF provides primitives to build a model of the original statement. The so-created model yields a new resource to which additional information can be attached; for example, metadata about a given triple. The properties that RDF provides to explicitly represent and describe triples without asserting them are given in [Table 4.7](#). This mechanism to make statements about statements is referred to as reification. Note that the RDF specification does not assign a normative formal semantics to the reification vocabulary and neither will it be developed here.

Table 4.6

RDF serializations of closed lists

<p>Turtle</p> <pre><http://example.org/doc.html> a exs:Report ; xs:chapters (<http://example.org/doc.html#chapter1> <http://example.org/doc.html#chapter2> <http://example.org/doc.html#chapter3>).</pre>
<p>RDF/XML</p> <pre><exs:Report rdf:about="http://example.org/doc.html"> <exs:chapters rdf:parseType="Collection"> <rdf:Description rdf:about="http://example.org/doc.html#chapter1"/> <rdf:Description rdf:about="http://example.org/doc.html#chapter2"/> <rdf:Description rdf:about="http://example.org/doc.html#chapter3"/> </exs:chapters> </exs:Report></pre>

The use of RDF reification is rather discouraged as the semantics of reification are unclear and reified statements are quite cumbersome. Although RDF provides the constructs to write reifications, in RDF asserting the reification is not the same as asserting the original statement, and neither implies the other. Moreover, reification expands the initial triple into a total of five triples (a triple plus a reification quad) and the link between the initial triple and its reification is not maintained. Metadata about statements should preferably be attached to the data source instead of the reified triple. Moreover, there is a strong movement to introduce the concept of graph and named graphs in RDF, which allows for statements about graphs that would not require reification at all. The interested reader is referred to the future issues section of this chapter for further insights on the named graph discussion at W3C.

Table 4.7

RDF reification

Triple <http://example.org/doc.html> a exs:Report.
Reified triple in Turtle syntax <pre> exr:triple1 a rdf:Statement ; rdf:subject <http://example.org/doc.html> ; rdf:predicate rdf:type ; rdf:object exs:Report.</pre>
<pre> graph LR exr_triple1([exr:triple1]) -- rdf:type --> rdf_statement([rdf:Statement]) exr_triple1 -- rdf:subject --> http_uri([http://example.org/doc.html]) exr_triple1 -- rdf:predicate --> rdf_type([rdf:type]) exr_triple1 -- rdf:object --> exs_report([exs:Report]) </pre>

4.2.4 Lightweight Ontology Formalization with RDFS

RDFS stands for RDF schema and is a semantic extension to RDF. It provides mechanisms for describing groups of related resources and the relationships between these resources (www.w3.org/TR/rdf-schema/). It is a lightweight language to declare and describe the resource types (called classes) and resource relationship and attribute types (called properties). RDFS allows one to name and define vocabularies used in labeling RDF graphs: naming the classes of existing resources; naming relation types existing between instances of these classes and giving their signatures, that is, the type of resources they connect. RDFS defines inferences to be applied using these hierarchies of types and the signatures of properties. Providing a URI for types, RDFS allows one to declare the taxonomic skeleton of an ontology in a universal language, with universal identifiers and semantics.

More formally, the semantics of RDFS is based on sets and set operators (union intersection and inclusion). Already RDF provides the membership declaration property (`rdf:type`) that allows one to capture in a graph structure which resource belongs to which class (a kind of set) and which couple of resources belong to which relation (another kind of set). RDFS provides the vocabulary to describe the relations between these sets: relations between classes, relations between properties, and between classes and properties.

4.2.4.1 Taxonomical Skeleton of Resource Classes

RDFS definitions factorize some of the information about the RDF data so that it is no longer needed to repeat that information. The information is no longer explicitly stated in

the data but can be derived through inferences. For instance, by saying that the class `Man` (a set of resources) is a subclass (subset) of the class `Person` (another set of resources) it is no longer required to say that `Fabien` is a `Person` and a `Man`. This can be derived from the fact that `Fabien` is a `Man` and the fact that `Man` is a subclass of `Person` so that `Fabien` is also a `Person`.

All objects described in RDF are per the definition of the type `rdfs:Resource` which itself is of type `rdfs:Class`. Given by the class hierarchy of RDFS, any class in RDF is a subclass of `rdfs:Resource`, and consequently an instance of type `rdfs:Class`. RDF moreover defines classes for typing literals as `rdfs:Literal` or datatypes as `rdfs:Datatype`. The particular class `rdf:XMLLiteral` is the class of XML literal values (in RDFS, XML literals are the only predefined datatype; all other datatypes, such as string or integers, but also more complex datatypes such as dates and time, have to be defined in an external vocabulary with own URIs and a corresponding function that maps the datatype URIs to actual datatype interpretations), is an instance of `rdfs:Datatype`, and a subclass of `rdfs:Literal`; in fact, all datatypes in RDFS are subclasses of the class of all literal values. For more details the interested reader is referred to the official specification of RDFS (www.w3.org/TR/rdf-schema/).

The same as for classes accounts for properties in RDF; they are first class citizens too and typed by means of the class `rdf:Property`. Classes and properties are organized in hierarchies via `rdfs:subClassOf` and `rdfs:subPropertyOf` statements. Both these properties define transitive and reflexive relationships between classes and properties, respectively.

As introduced earlier, `rdf:type` is the property used to state that a resource is an instance of a class, and all its super-classes too. The type of a resource propagates through the hierarchy defined by `rdfs:subClassOf`. In contrast to object-oriented languages such as Java, the property `rdfs:subClassOf` allows multiple inheritance. The example in [Table 4.8](#) introduces a class `Document` that has several children/subclasses and hence includes the union of `Report`, `Advert`, and `Presentation`. The most specific class `PresentationReport` has several parents which are `Report` and `Presentation` and can be seen to include the union of `Report` and `Presentation`.

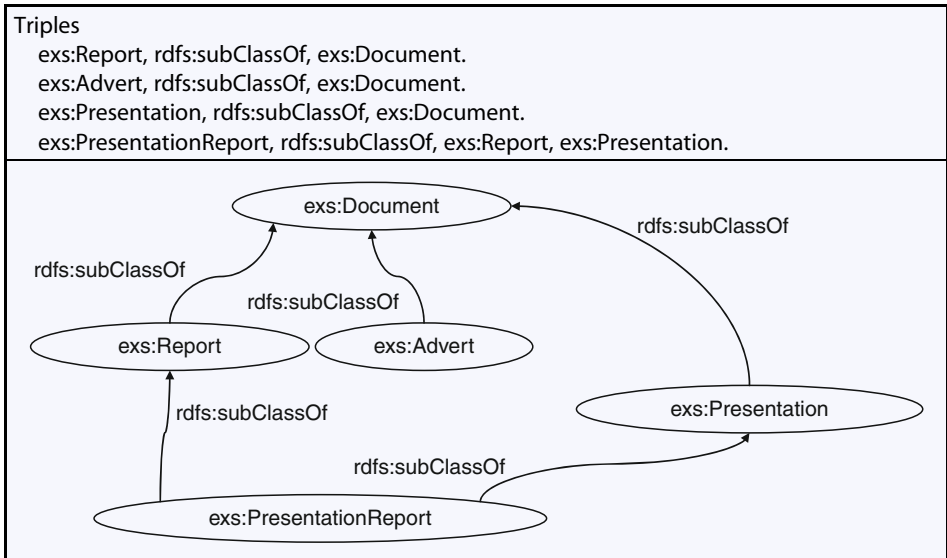
4.2.4.2 Taxonomical Skeleton of Resource Relations

As stated above, by means of `rdfs:subPropertyOf` statements, it is possible to define hierarchies in the same way as with classes; for example, a `hasAuthor` property could be seen as a subtype of a `hasCreator` property and inherits its signature.

RDFS furthermore extends the signature of a property definition so that the classes of the resources that are linked by a property can be specified. The subject type of a triple is termed the domain of a property, while the range defines the type of the object. The terms domain and range were borrowed from mathematics where the domain of a function is the set of values for which it is defined and the range is the set of values it can take. The property `rdfs:domain` is used to determine the domain, and moreover to state that any resource that has a given property is an instance of a given class. For example ([Table 4.9](#)), if the property `foaf:name` is related to the class `foaf:Person` via `rdfs:domain`, any resource such as

Table 4.8

RDFS class hierarchies



`exr:JohnMayers`, then, per RDFS semantics, `exr:JohnMayer` is of type `foaf:Person`; although without the explicit claim that any researcher is a person. The property `rdfs:range` is the equivalent for the ranges so that an RDF property relates resources belonging to its domain to resources belonging to its range. In terms of RDFS semantics, the same deductions occur: resources in the range of a property assume the class membership specified by the `rdfs:range` description of the given property. When several domains or ranges are given, the instances belong to all the given classes. The signature of properties allows one to type data through their usage: every time a property is used the resources it links will be typed using its domains and its ranges. To this end, properties and their definitions are really important since they capture the semantics of the resources that are linked.

To summarize the presentation of RDFS, [Fig. 4.3](#) depicts the entire RDF(S) vocabulary and relationships, and provides a concluding example. A broken curved line stands for an `rdf:type` relation, and a solid straight line represents an `rdfs:subClassOf` relation. The figure also shows that `rdfs:Resource` is a superclass of all other classes, and `rdfs:Class` is a class of all classes, including `rdfs:Class` itself. A class of classes is called *metaclass* in CLOS (Common Lisp Object System). Consequently, `rdfs:Class` and `rdfs:Datatype` in RDFS vocabulary are similar to metaclasses in CLOS. Properties such as `rdfs:seeAlso`, `rdfs:isDefinedBy` or `rdfs:label`, and `rdfs:comment` are used to provide additional information about classes and instances in an RDFS vocabulary, such as related resources (`rdfs:seeAlso`) or natural language tags (`rdfs:label`) for human consumption. For a more detailed description of these and other entities in [Fig. 4.3](#), the reader is referred to the official W3C pages for RDF and RDFS (www.w3.org/RDF/; www.w3.org/TR/rdf-schema/).

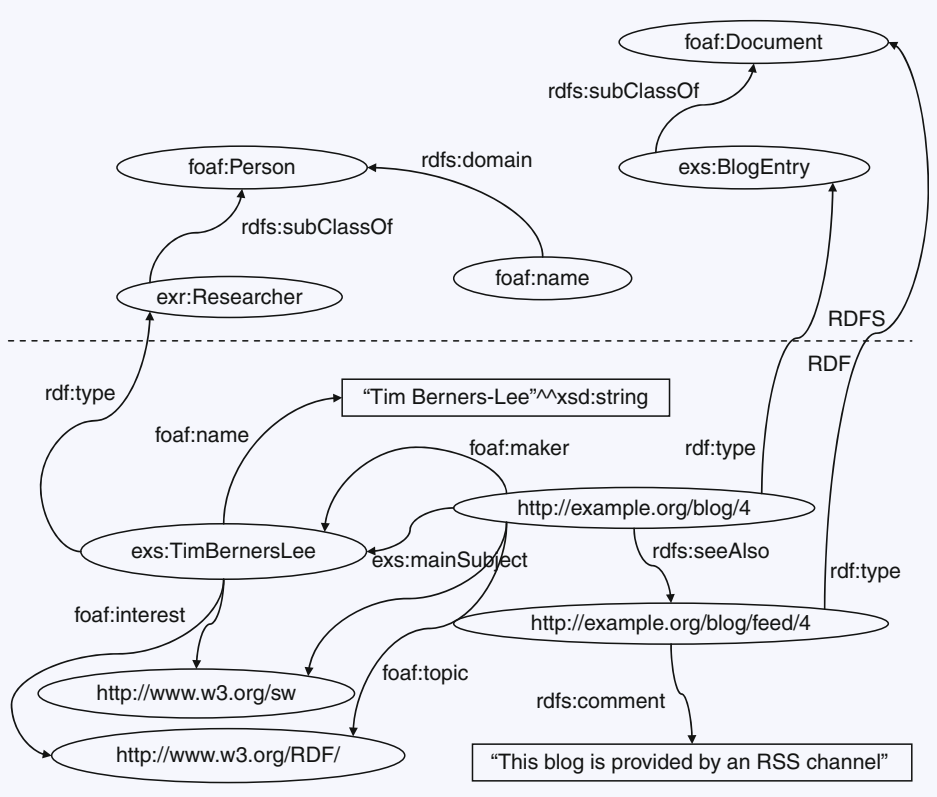
Table 4.9

Comprehensive RDF(S) example

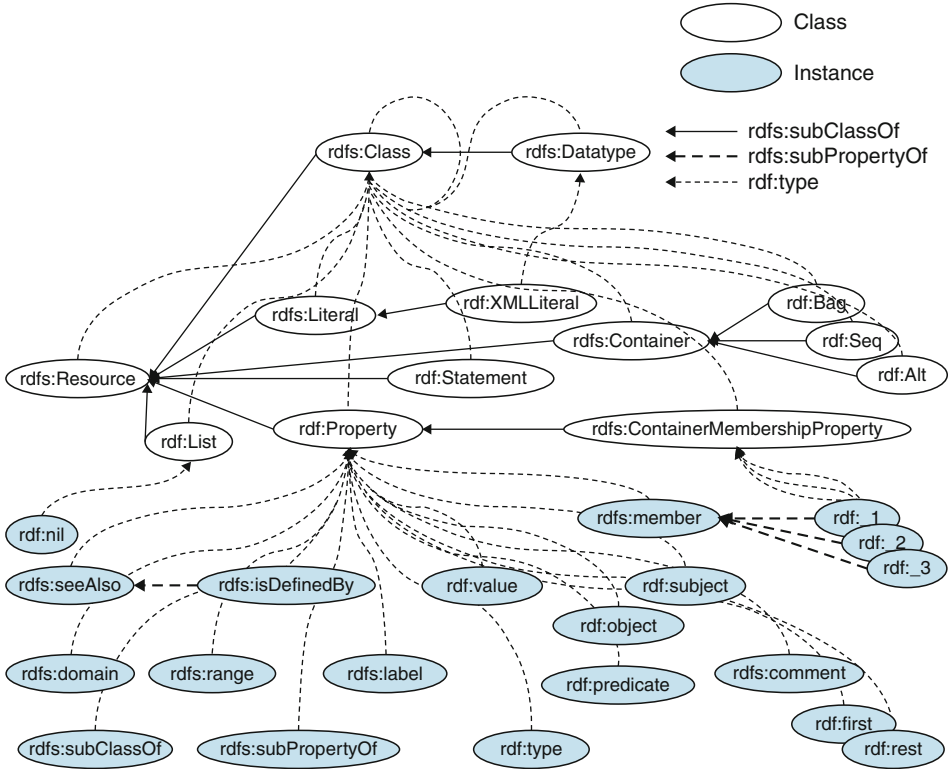
Triples

```

exs:Researcher rdfs:subClassOf foaf:Person.
foaf:name rdfs:domain foaf:Person.
exr:JohnMayers a exr:Researcher ;
    foaf:name "John M. Mayers" ;
    foaf:interest <http://www.w3.org/sw/>, <http://www.w3.org/RDF/>.
exs:BlogEntry rdfs:subClassOf foaf:Document.
<http://example.org/blog/4> a exs:BlogEntry ;
    exs:mainSubject exr: JohnMayers;
    rdfs:maker, exr: JohnMayers;
    foaf:topic <http://www.w3.org/sw/>,
        <http://www.w3.org/RDF/> ;
    rdfs:seeAlso <http://example.org/blog/feed/4>.
<http://example.org/blog/feed/4> a exs:Document ;
    rdfs:comment "This blog is provided by an RSS channel."
  
```



RDFS was defined in RDF, and hence RDFS itself and the schemas defined with RDFS are specified by means of RDF graphs and triples. To this end, any RDF-minded tool can be used straightforwardly to manipulate schemas too, and in particular the SPARQL query



■ Fig. 4.3

RDFS hierarchy graph (RDF lecture: www-asm.nii.ac.jp/~koide/SWCLOS2/Manual/RDFSchem.html)

language can be used for querying both data and their schemas. ▶ [Table 4.9](#) provides concluding examples leveraging RDFS through RDE, and showcasing how the two formalisms can be used to build one integrated graph. First of all, the RDF(S) example defines any researcher to be a person, and states that the domain of the property foaf:name is such a person. Furthermore, the third triple indicates that John Mayers is a researcher whose interests are the Semantic Web and RDF. The same John Mayers is the subject and maker of the weblog entry number 4 on the topics Semantic Web and RDF. The entry by John Mayers refers to an RSS feed for the blog, which itself is a document, but not a weblog entry.

4.2.5 Limitations of RDF/S in Building a Logical Framework

RDF is a simple yet powerful data model and language for describing Web resources. Using RDF one can make statements about Web resources in the form of triples (`<subject, predicate, object>`), triples that can be combined in large graph structures. An RDF graph can be seen as a collection of facts where each triple represents

a ground fact if the edges are URIs or literals, or existential quantified binary predicates, where existential qualified variables correspond to blank nodes. RDF schema (RDFS) extends RDF by introducing means to model classes, property hierarchies of classes and properties, as well as simple domain and range restrictions.

Using the graph or triple-based model gives RDF(S) a very high flexibility but at the same time makes it an unstable foundation for layering logic-based languages on top [5]. The major source of this problem is that in RDF(S) one can make arbitrary statements over statements and by consequence there is not restriction in mixing the data and metadata levels. For example, the statement `<rdfs:Class rdf:type rdfs:Class>` states that `class` is an instance of a `class`. Users can change thus the semantics of the ontology modeling constructs. In logic languages such as OWL-Lite or OWL-DL, the two levels are clearly separated, statements in the language being separated from statements about the language.

The extra features of RDF(S) mentioned before, for example, the usage of language constructs as part of the language vocabulary, create obstacles when layering logical languages, that is, OWL-Lite and OWL-DL on RDF(S). The problem of layering languages on top of RDF(S) has been pointed out since the initial design of OWL [6] and are also covered in the Introductory Chapter of this text.

Another issue that hampers the proper layering and furthermore the integration of Semantic Web languages is the different assumptions these languages adhere to, that is, closed-world or open-world assumptions. In a closed world any missing information is considered as false, while in an open world missing information is treated as unknown. Furthermore, in languages of systems that adhere to the closed-world assumption, the schema information behaves as constraints over the data. More precisely, the constraints need to be satisfied by the instance data in order to have a consistent model. Most procedural programming languages, relational databases, and rules-based systems make the closed-world assumption. In the case of relation databases for example, legal database states are defined using integrity constraints. What would trigger a constraint violation in a closed-world assumption system leads to entailment of new information in systems that adhere to open-world assumption. In this case the schema is used to infer new knowledge, that is, to promote as implications. To illustrate the differences between the two assumptions let us consider the following set of statements expressed in predicate logic:

```
Person(John)
Book(SemanticWebTextBook)
 $\forall x, \forall y \text{ hasAuthor}(x, y) \rightarrow \text{Book}(x) \wedge \text{Author}(y)$ 
```

If a new statement is added,

```
hasAuthor(SemanticWebTextbook, John)
```

to the previous theory, a different behavior shall be noticed for systems making the closed-world versus open-world assumption. In the case of closed-world assumption, a constraint violation is triggered as John is a Person but not an Author. In the case of open-world assumption a new fact is inferred, namely that John is an Author; that is, `Author(John)`.

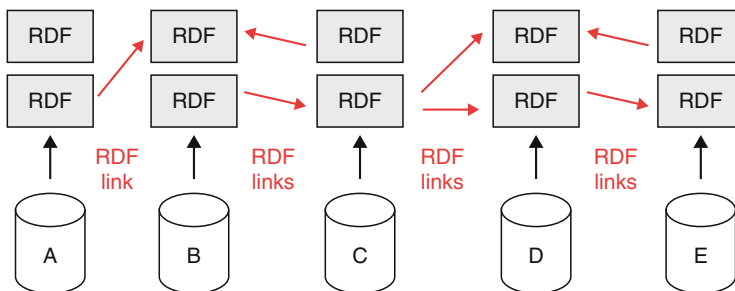
RDF adheres to open-world assumption. At a first glance this seems to be the right decision as the Web is an open environment. On the other hand, one can argue that the closed-world assumption is more appropriate when reasoning on a portion of the Web. Which assumption is most suitable for the Web remains to be proven. A wise approach is to stay agnostic and to leave at the application level the decision under which assumption the information is interpreted. This is the approach taken by RDFS.

4.3 Examples and Applications

4.3.1 Data Integration on the Web

By being the de facto standard for metadata and for annotating data on the Web, RDF has definitively emerged to become the premier data model for the integration and interoperability of heterogeneous datasets. RDF provides all the technicalities needed for interlinking distributed data sources at the data instances level as much as at the data annotation level. This trend cumulated in the linked data initiative (<http://linkeddata.org/>) whose goal it is to enable people to share structured data on the Web as easily as they can share documents and multimedia today via the World Wide Web. Linked data are about connecting related datasets that were not previously linked, or using basic Semantic Web technology such as RDF/S to lower the entry barriers to the publishing and interlinking of datasets that currently reside in silos or that are linked using other non-Web-oriented methods (► Fig. 4.4). RDF/S has thus application to any information portal, data-intensive website or data integration project, as its flexibility and extensibility allows for the opening up of domain-specific data silos and databases; including relationally specified data. To this end, RDF/S is of high interest for creating cross-domain datasets and for fostering a networked economy.

In this context, the linked data movement releases best practices of how to publish data, and promotes the use of RDF to make any type of potentially closed-in data available publicly on the Web, namely through the use of URIs to denote things, and the use of HTTP to look up information about these things. The information could be given back as human-consumable HTML-renderings of the description, or as RDF dumps or SPARQL endpoints.



■ Fig. 4.4
Concept of linked data [2]

Additionally, the linked data initiative promotes the interlinking of datasets by means of shared URIs which make the linked datasets a browsable asset, again, just as HTML documents are interlinked and browsable goods on the human-targeting World Wide Web. To this end, the Web of Data has developed to become the most impacting and visible use case of RDF/S, so far. Essentially, the Web of Data is finally in the process of realizing the Semantic Web, as it was intended to be from the very early days [1]; and in this context RDF has become the fundamental layer of the Semantic Web, also in practice. More details about the linked data initiative and the Web of Data are provided in [▶ Semantic Annotation and Retrieval: Web of Data](#).

4.3.1.1 Schema and Data Samples

As discussed in the introduction to this section, the Web of data is about interlinking datasets, which essentially results in the interlinking of documents on the Web, just as with hypertext documents too. However, unlike in the Web of hypertext, where links are relationship anchors in hypertext documents written in HTML, for data, the links are established between arbitrary things and graphs described in RDF.

This section presents some prominent examples of datasets and schemas that were recently made publicly available via RDF/S, or that are used to make data silos available according to linked data principles.

- A. *Documents* (e.g., by means of the Dublin Core vocabulary – www.dublincore.org): Annotation of title, author, creation date, or subject among many other properties, for instance, to make digital library content usable to applications on the Web, such as cross-library search engines.

Schema Examples:

- <http://purl.org/dc/terms/Agent> is a Class: A resource that acts or has the power to act
- <http://purl.org/dc/terms/contributor> is a Property: An entity responsible for making contributions to the resource with the Range: <http://purl.org/dc/terms/Agent>
- <http://purl.org/dc/terms/creator> is a Property: An entity primarily responsible for making the resource that is SubPropertyOf: <http://purl.org/dc/terms/contributor> and that has as Range: <http://purl.org/dc/terms/Agent>
- <http://purl.org/dc/terms/created> is a Property: Date of creation of the resource that is SubPropertyOf: <http://purl.org/dc/terms/date> with the Range: <http://www.w3.org/2000/01/rdf-schema#Literal>
- <http://purl.org/dc/terms/isReplacedBy> is a Property: A related resource that displaces, or supersedes the described resource that is SubPropertyOf: <http://purl.org/dc/terms/relation>

There is a document Document1 with three contributors, whereof one is the main creator. The document is a DCMI Metadata example and was created on June 15, 2009; subsequently, it was replaced by a new document referred to as Document2.


```

@prefix dc: <http://purl.org/dc/terms/>.
@prefix ex: <http://www.example.org/>.
ex:Document1 dc:contributor ex:Contributor2, ex:Contributor3;
    dc:creator ex:Contributor1;
    dc:created "2009-06-15";
    dc:description "This is a DCMI Metadata example";
    dc:isReplacedBy ex:Document2.

```

- B. *Persons* (e.g., FOAF – www.foaf-project.org; VCard – www.w3.org/Submission/vcard-rdf/): Annotations of people descriptions with name, e-mail, Web page, and links to friends are used to make personal websites intelligible to browsers, and offer services to the user like saving contact details in his address book.

Schema Example:

- <http://xmlns.com/foaf/0.1/Person> is a Class: A person and SubClassOf: <http://xmlns.com/foaf/0.1/Agent>
- <http://xmlns.com/foaf/0.1/name> is a Property: A name for something with Range: <http://www.w3.org/2000/01/rdf-schema#Literal>
- <http://xmlns.com/foaf/0.1/workplaceHomepage> is a Property: The homepage of an organization a person works for with Domain: <http://xmlns.com/foaf/0.1/Person> and Range: <http://xmlns.com/foaf/0.1/Document>
- <http://www.w3.org/2006/vcard/ns#VCard> is a Class: An electronic business card
- <http://www.w3.org/2006/vcard/ns#fn> is a Property: The full name of the object the vCard represents with Domain: <http://www.w3.org/2006/vcard/ns#VCard> and Range: <http://www.w3.org/2000/01/rdf-schema#Literal>
- <http://www.w3.org/2006/vcard/ns#title> is a Property: Job title, functional position or function of the object the vCard represents with Domain: <http://www.w3.org/2006/vcard/ns#VCard> and Range: <http://www.w3.org/2000/01/rdf-schema#Literal>

There is some person with the name Peter Example who works as System Administrator at the company whose website is www.example.org/myCompany.

```

@prefix vc: <http://www.w3.org/2006/vcard/ns#>.
@prefix foaf: <http://xmlns.com/foaf/0.1/>
[] a foaf:Person;
    foaf:name "Peter Example";
    foaf:workplaceHomepage <http://www.example.org/myCompany>;
    vc:title "System Administrator".

```

- C. *Organizations* (FOAF; GoodRelations – www.purl.org/goodrelations/): Activities, public/private, name, branches, domain for instance to automatically building and maintaining yellow pages services.

Schema Example:

- <http://purl.org/goodrelations/v1#BusinessEntity> is a Class: The legal agent making a particular offering

- <http://purl.org/goodrelations/v1#legalName> is a Property: The legal name of the business entity with Domain: <http://purl.org/goodrelations/v1#BusinessEntity>

There is some business entity with the legal name “The Example Company Ltd.” that is located at Example Road 2 in 12345 ExampleCity. The company’s email address is office@example-company.org

```
@prefix vc: <http://www.w3.org/2006/vcard/ns#>.
@prefix gr: <http://purl.org/goodrelations/v1#>
[] a gr:BusinessEntity;
   gr:legalName "The Example Company Ltd. ";
   vc:email <mailto:office@example-company.org>;
   vc:adr [vc:street "Example Road 2 ;
             vc:locality "ExampleCity" ;
             vc:postal-code "12345" ].
```

- D. *Copyrights* (e.g., Creative Commons – www.creativecommons.org/ns): License and conditions for the reuse of digital works to filter search results to those that can actually be used for one’s task.

Schema Example:

- <http://creativecommons.org/ns#Work> is a Class: A potentially copyrightable work
- <http://creativecommons.org/ns#License> is a Class: A set of requests/permissions to users of a Work
- <http://creativecommons.org/ns#license> is a Property: A Work has a license and SubPropertyOf: <http://purl.org/dc/terms/license> with Domain: <http://creativecommons.org/ns#Work> and Range: <http://creativecommons.org/ns#License>
- <http://creativecommons.org/ns#legalcode> is a Property: The URL of the legal text of a License with Domain: <http://creativecommons.org/ns#License>

There is a piece of work with an Attribution-ShareAlike 3.0 license from creative commons that allows distribution of the work under the condition that the owner is attributed. The human-readable legal text of the license is published at <http://creativecommons.org/licenses/by-sa/3.0/legalcode>.

```
@prefix cc: <http://creativecommons.org/ns#>.
@prefix ex: <http://www.example.org/>.
ex:myWork cc:license
    <http://creativecommons.org/licenses/by-sa/3.0/>.
<http://creativecommons.org/licenses/by-sa/3.0/>
  cc:permits <http://web.resource.org/cc/Distribution>;
  cc:requires <http://web.resource.org/cc/Attribution>;
  cc:legalcode
    <http://creativecommons.org/licenses/by-sa/3.0/legalcode>.
```

- E. *Products* (GoodRelations; eClass – www.ebusiness-unibw.org/ontologies/eClass): Prices, references, reviews, availability, shipping, etc., for instance, to customize catalogs or aggregate feedbacks into benchmarks. GoodRelations has recently been adopted by Google.

Schema Example:

- <http://purl.org/goodrelations/v1#hasUnitOfMeasurement> is a Property: The unit of measurement given using the UN/CEFACT Common Code with Domain: <http://purl.org/goodrelations/v1#QuantitativeValue>, and <http://purl.org/goodrelations/v1#UnitPriceSpecification> and Range: <http://www.w3.org/2001/XMLSchema#string>
- <http://purl.org/goodrelations/v1#hasPriceSpecification> is a Property: This links an offering to one or more price specifications with Domain: <http://purl.org/goodrelations/v1#Offering> and Range: <http://purl.org/goodrelations/v1#DeliveryChargeSpecification>, <http://purl.org/goodrelations/v1#PaymentChargeSpecification>, <http://purl.org/goodrelations/v1#PriceSpecification>, and <http://purl.org/goodrelations/v1#UnitPriceSpecification>

There is a product of type pencil [AKF303003] whose length [BAF559001] is 150 mm.

@prefix eco:

<<http://www.ebusiness-unibw.org/ontologies/eClass/5.1.4/#>>.

@prefix gr: <<http://purl.org/goodrelations/v1#>>.

@prefix xsd: <<http://www.w3.org/2001/XMLSchema#>>.

[] a eco:C_AKF303003-gen ;

eco:P_BAF559001 [a gr:QuantitativeValueFloat ;

gr:hasUnitOfMeasurement "MMT"^^xsd:string ;

gr:hasValueFloat "150.0"^^xsd:float.

- F. *Calendar/Events* (RDF Calendar – www.w3.org/2002/12/cal/; NCAL – www.semanticdesktop.org/ontologies/ncal/): Name, dates, location, or durations to allow various calendar data to be integrated, imported, and shared in Web applications.

Schema Example:

- <http://www.w3.org/2002/12/cal/ical#Vevent> is a Class: A grouping of component properties that describe an event
- <http://www.w3.org/2002/12/cal/ical#location> is a Property: Defines the intended venue for the activity defined by a calendar component with Domain: <http://www.w3.org/2002/12/cal/ical#Vevent>, and <http://www.w3.org/2002/12/cal/ical#Vtodo>, and Range: <http://www.w3.org/2001/XMLSchema#string>
- <http://www.w3.org/2002/12/cal/ical#dtstart> is a Property: Specifies when the calendar component begins with Domain: <http://www.w3.org/2002/12/cal/ical#Vevent>, <http://www.w3.org/2002/12/cal/ical#Vtodo>, and <http://www.w3.org/2002/12/cal/ical#Vfreebusy> and Range: <http://www.w3.org/2001/XMLSchema#dateTime>

- <http://www.w3.org/2002/12/cal/ical#dtend> is a Property: Specifies the date and time that a calendar component ends with Domain: <http://www.w3.org/2002/12/cal/ical#Vevent>, <http://www.w3.org/2002/12/cal/ical#Vfreebusy> and Range: <http://www.w3.org/2001/XMLSchema#dateTime>

There is an iCal event planned for to take place at the Example Hotel in London on June 24, 2010 between 9 am and 6 pm.

```
@prefix cal:    < http://www.w3.org/2002/12/cal/ical#>.
[] a cal:Vevent;
   cal:location "Example Hotel, London, UK";
   cal:dtstart  "2010-06-24T09:00:00";
   cal:dtend    "2010-06-24T18:00:00".
```

- G. *Places* (GeoNames – www.geonames.org/ontology/; WGS84 – www.w3.org/2003/01/geo/wgs84_pos): Geographical locations, coordinates, countries to combine geo data with mapping views or use location and places as filtering criterion in Web searches.

Schema Example:

- <http://www.geonames.org/ontology#Feature> is a Class: A geographical object uniquely defined by its geonames id
- <http://www.geonames.org/ontology#name> is a Property: The preferred name of the feature with Domain: <http://www.geonames.org/ontology#Feature>
- <http://www.geonames.org/ontology#countryCode> is a Property: A two letters country code in the ISO 3166 list with Domain: <http://www.geonames.org/ontology#Feature>
- http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing is a Class: Anything with spatial extent
- http://www.w3.org/2003/01/geo/wgs84_pos#lat is a Property: The WGS84 latitude of a SpatialThing with Domain: http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing
- http://www.w3.org/2003/01/geo/wgs84_pos#long is a Property: The WGS84 longitude of a SpatialThing with Domain: http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing

The resource with geonames ID 2643743 represents London in the UK, given by the ISO country code GB. The geographic location is given by the WSG84 latitude/longitude coordinates 51.5005/-0.1288.

```
@prefix geo:    <http://www.geonames.org/ontology#>.
@prefix wsg:    <http://www.w3.org/2003/01/geo/wgs84_pos#>.
<http://sws.geonames.org/2643743/>
    geo:name "London" ;
    geo:countryCode "GB" ;
    wsg:lat "51.5005149421307" ;
    wsg:long "-0.12883186340332".
```

- H. *Social networks* (SIOC – www.sioc-project.org; RELATIONSHIP– vocab.org/relationship/): Go beyond the silo architecture and allow connections and interoperability across Web 2.0 platforms.

Schema Example:

- <http://rdfs.org/sioc/ns#Community> is a Class: A high-level concept that defines an online community and what it consists of
- <http://rdfs.org/sioc/ns#Post> is a Class: An article or message that can be posted to a Forum and SubClassOf: <http://rdfs.org/sioc/ns#Item>
- http://rdfs.org/sioc/ns#has_creator is a Property: This is the User who made this Item with Domain: <http://rdfs.org/sioc/ns#Item> and Range: <http://rdfs.org/sioc/ns#User>
- <http://rdfs.org/sioc/ns#topic> is a Property: A topic of interest; for example, in the Open Directory Project or of a SKOS category and SubPropertyOf: <http://purl.org/dc/terms/subject>
- http://rdfs.org/sioc/ns#has_reply is a Property: Points to an Item that is a reply or response to this Item with Domain: <http://rdfs.org/sioc/ns#Item> and Range: <http://rdfs.org/sioc/ns#Item>

There is a post from September 7, 2006 by the author identified by <example.org/author> on the topic of annotation. There is a comment in reply to this post.

```
@prefix sioc:<http://rdfs.org/sioc/ns#>.
@prefix dc:<http://purl.org/dc/terms/>.
<http://example.org/blog/2010/entry> a sioc:Post ;
    dc:created "2006-09-07T09:33:30Z" ;
    sioc:has_creator <http://example.com/author/> ;
    sioc:topic <http://example.org/topics/annotation> ;
    sioc:has_reply          <http://example.org/blog/2010/
entry#comment-1> .
```

- I. *Lexicons, Indexes, Folksonomies* (SKOS – www.w3.org/2004/02/skos/, NiceTag – <http://ns.inria.fr/nicetag/2010/07/21/voc>): Semantic descriptions of thesauri, classification schemes, subject heading lists, and taxonomies.

Schema Example:

- <http://www.w3.org/2004/02/skos/core#Concept> is a Class: A concept in a knowledge organization system
- <http://www.w3.org/2004/02/skos/core#OrderedCollection> is a Class: An ordered collection of concepts
- <http://www.w3.org/2004/02/skos/core#prefLabel> is a Property: The preferred lexical label for a resource, in a given language and SubPropertyOf: <http://www.w3.org/2000/01/rdf-schema#label>
- <http://www.w3.org/2004/02/skos/core#narrower> is a Property: Relates a concept to a concept that is more specific in meaning

- <http://www.w3.org/2004/02/skos/core#broader> is a Property: Relates a concept to a concept that is more general in meaning
- <http://www.w3.org/2004/02/skos/core#related> is a Property: Relates concepts for which there is an associative semantic relationship
- <http://ns.inria.fr/nicetag/2010/07/21/voc#AnnotatedResource> is a Class: A dereferenceable resource on the Web that is taggable
- <http://ns.inria.fr/nicetag/2010/07/21/voc#isRelevantToSt> is a Property: Links a resource to anything that it may be relevant to

The topic annotation is related to the concepts metadata and footnote, where footnote is a narrower term than annotation. The annotation concept is moreover described as being relevant to the book chapter on RDF/S.

```
@prefix skos:    <http://www.w3.org/2004/02/skos/core#>.
@prefix voc:    <http://ns.inria.fr/nicetag/2010/07/21/voc#>.
<http://example.org/topics/annotation> a skos:Concept;
    skos:definition "Any descriptive notation applied to data";
    skos:prefLabel "Annotation";
    skos:narrower <http://example.org/topics/footnote>;
    skos:related <http://example.org/topics/metadata>;
    a voc:AnnotatedResource;
    voc:isRelevantToSt <http://example.org/bookchapter/RDFS>.
```

- J. *Genetics and Life Science* (e.g., Gene Ontology – www.geneontology.org; Open Biomedical Ontology – www.obofoundry.org; OMIM – www.ncbi.nlm.nih.gov/omim): Molecular functions, biological processes, cellular components, and vocabularies to describe the human anatomy and genes, biochemistry, genetic disorders or phenotypes.

Schema Example:

- <http://www.geneontology.org/dtd/go.dtd#term> is Class: Any term in the gene ontology is of type term
- http://www.geneontology.org/dtd/go.dtd#is_a is a Property: If A is_a B, then A is a subtype of B with Domain: <http://www.geneontology.org/dtd/go.dtd#term> and Range: <http://www.geneontology.org/dtd/go.dtd#term>
- http://www.geneontology.org/dtd/go.dtd#part_of is a Property: Representation of part–whole relationships
- <http://www.geneontology.org/dtd/go.dtd#regulates> is a Property: One process directly affects the manifestation of another process or quality

The term GO:0000001 represents the concept of mitochondrion inheritance which is a subtype of the concept GO:0048308 (organelle inheritance), a part of the concept GO:0009530 (primary cell wall), and negatively regulates GO:0006312 (mitotic recombination).

```
@prefix godtd: <http://www.geneontology.org/dtd/go.dtd#>.
@prefix go: <http://www.geneontology.org/go#>.
```

```

go:GO:0000001 a go:term ;
  godtd:accession "GO:0000001" ;
  godtd:name "mitochondrion inheritance" ;
  godtd:synonym "mitochondrial inheritance" ;
  godtd:definition "The distribution of mitochondria, including the mitochondrial genome, into daughter cells after mitosis or meiosis, mediated by interactions between mitochondria and the cytoskeleton."
  godtd:is_a go: GO:0048308 ;

```

In principle the string text should be on the same line as the `godtd:definition` attribute. Assuming that the text does not fit one line, it should be broken as given above.

“The distribution of mitochondria, including the mitochondrial genome, into daughter cells after mitosis or meiosis, mediated by interactions between mitochondria and the cytoskeleton.”

```

  godtd:is_a go:GO:0048308 ;
  godtd:part_of go:GO:0009530 ;
  godtd:negatively_regulates go:GO:0006312.

```

Healthcare (SNOMED CT – www.ihtsdo.org/snomed-ct/; MeSH – www.nlm.nih.gov/mesh/; UMLS – www.nlm.nih.gov/research/umls/): Clinical and disease-related aspects, medications, treatments and health records for the integration of patient records, and the optimization of medical information flows.

4.3.2 RDF Web Application

Although the listing above presents mostly independent vocabularies and datasets, the strength of RDF lies in the flexibility of integration. RDF provides, in contrast to relational data models, a schema-last data model. RDF graphs can in principle quite easily be merged by sharing particular resources, or claiming two resources to be the same, although their identifier might be different. The integration approach that RDF offers is driven by the linking of resources in the subjects and objects of triples. In fact, as discussed previously, linking resources by means of URIs is one of the principles of the linked data initiative. Reconsidering the listing above, RDF allows, for example, to link the description of a document to the description of its author (person), linking the author to the description of her affiliation (organization) and colleagues, linking the organization to the description of the geographical location that is linked to transportation plans, hotels, events and the history, and demographics of the place; from the history there would be links to important personalities which are linked to events, groups, and biographical data, and so on.

RDF not only provides this ability to publish and link the data, it also provides the foundational shared data model on which other capabilities are built: querying these data

(SPARQL is built on top of RDF), embedding (RDFa and GRDDL rely on the RDF model), and inference and reasoning (RDFS and OWL are, at least partly, defined on top of RDF, even RIF has a part dealing with RDF). While details about such related technologies and languages are subject to the respective chapters in this book, this section concludes with a review of some well-known and representative applications that are built on top of and/or using RDF/S.

The BBC website (www.bbc.co.uk) creates Web identifiers for every item to enable very rich cross-domain aggregation of information. The content that is collected in a comprehensive dataset can be discovered by users in many different ways, via the website, the API, or dedicated query engines. Essentially, the RDF representations allow developers to use BBC data to build richer applications, including new BBC products, as the Web portal has evolved to become an API. The development of applications that use BBC data is now extremely loosely coupled, as the content is independent of any system constraints or data formats. As an example, a Web identifier is provided for every species, habitat, and adaptation mentioned in the BBC programs. Data are aggregated from Wikipedia, the Animal Diversity Web, WWF's Wildfinder, the Zoological Society of London's EDGE of Existence program, and the IUCN's Red List of Threatened Species. These data are then reintegrated and linked out to multimedia resources from the BBC to jointly provide a highly comprehensive information portal about the BBC program and the shown content.

In May 2008, Yahoo! launched SearchMonkey (<http://developer.yahoo.com/searchmonkey/>) an open platform for search that allows developers to build applications on top of the Yahoo! search engine. In particular, SearchMonkey allows site owners to share structured data in the form of RDF/S and RDFa with the engine in order to customize and enhance the presentation of search results, and to implement applications on top of the search engine. For instance, a result found on LinkedIn will include name, surname, and position of a person; a result found on YouTube will include the title and a direct access to the video; a result found on Amazon will include the title, the author, and the average review. All of these results can be integrated, and thanks to RDF a far more sophisticated and interlinked search result experience can be created.

Open Calais from Thomson Reuters is a Semantic Web Service and API (<http://www.opencalais.com/>) that identifies entities within documents submitted to the service, and that extracts and annotates those in RDF. As such, Open Calais automatically creates rich semantic metadata for otherwise non-annotated content by means of natural language processing, machine learning, and other methods. The annotated things in documents provide the ability to connect documents with people, maps, and enterprises to relate companies to goods or events to locations. By leveraging the metadata, a large number of functionalities can be supported and improved: in particular, more precise searching, subject monitoring, contextual syndication, custom alerts and notification, the thematic routing of information, intelligence, link suggestion, and augmented browsing on the fly. More on automatic annotation will be discussed in [▶ Semantic Annotations and Retrieval: Manual, Semiautomatic, and Automatic Generation](#).

Zemanta (www.zemanta.com) is an API leveraging RDF and used in blogging tools. The Zemanta service supports the sharing of content, the retrieval of related contents,

■ **Table 4.10**

DBpedia example extract of the RDF entry

```
<http://dbpedia.org/resource/Resource_Description_Framework>
  rdfs:label "Resource Description Framework", "RDF" ;
  owl:sameAs umbel:Resource_Description_Framework ;
  skos:subject category:World_Wide_Web_Consortium_standards ;
  foaf:homepage <http://www.w3.org/TR/rdf-primer/> ;
  a yago:W3CStandards ;
  dbpprop:baseStandards dbpedia:XML, dbpedia:URI ;
  ...
```

images, and the automatic linking of those to related online resources. In principle, Zemanta, similarly to Open Calais with documents, helps in bringing blog entries in context, and in enhancing the user experience by automatically interlinking text with related knowledge and multimedia content. The same API is also used by other services including the semantic bookmarking service Faviki (www.faviki.com). In short, Faviki, as the hidden name reveals (favorites & Wikipedia), allows the connection of bookmarking tags and favorites to concepts listed by Wikipedia and DBpedia (www.dbpedia.org). DBpedia is an effort affiliated with the linked data initiative to extract structured data from Wikipedia, to make it available on the Web in form of RDF datasets, and to link other datasets to Wikipedia data – as done by Faviki. ▶ [Table 4.10](#) gives a short example of the DBpedia dataset for the Wikipedia entry of the Resource Description Framework. In the context of Zemanta and Faviki, the benefits of RDF are again eminent in automatically enriching bookmarking information and interlinking it by sharing tags via the URIs exposed through DBpedia.

Sindice (www.sindice.com) provides a Semantic Web index and search engine that crawls and indexes hundreds of million Web pages with RDF, RDFa, and Microformats in it and makes the data searchable and reusable. Examples of such data would be all different types of public resources that were presented in the previous section: profiles, contacts, calendar entries, events, social networks, reviews, and biographies. There are no limits in the domains covered, as long as the data are available publicly in any of the supported data formats. Sindice is one of the most sophisticated platforms for processing and querying linked data via a coherent set of functionalities and services.

MuseoSuomi (www.museosuomi.fi) is an application which publishes data on cultural collections. Through semantic technology, museums are able to consolidate their content and to provide visitors with enriched content-based search and browsing services over collections and databases that reach beyond their own availabilities and data sources. Similar ideas are followed by the CHIP project (www.chip-project.org) that applied semantic technologies to enrich the Rijksmuseum vocabularies and provide more expressive browsing, searching, and recommendation services. Additionally, the CHIP project leveraged annotations to personalize users' experiences both on the museum website and in the museum itself. The use of semantic technologies and RDF in particular is again to relate available artifacts with related information, namely the biographies of the

artist, the historical background, and similar pieces of art in terms of artist, period, or sociocultural similarities. By integrating the databases from various museums, a pan-site experience can be created for customers and visitors.

KmP (www-sop.inria.fr/acacia/soft/kmp.html) is a real-world experiment on the design and use of a customizable Semantic Web server to generate up-to-date views of the Telecom Valley of Sophia Antipolis in France. It supports the management of competencies at the level of organizations such as companies, research institute and labs, clubs, associations, government agencies, schools, and universities. The KmP platform aims at increasing the portfolio of competencies, actors, and projects of the technological pole of Sophia Antipolis by implementing a public knowledge management solution at the scale of the Telecom Valley based on a shared repository and a common language to describe and compare the needs and the resources of all the organizations. The project resulted in a portal that relies on a public Semantic Web server that is available for all the actors of the value chain. The KmP platform relies on RDF for the integration of heterogeneous and distributed data, for the querying from diverse viewpoints, for the adaptation of content to particular user needs, and for the general analysis, grouping, inferencing, and rendering of relevant indicators.

Semantic Web Pipes (pipes.deri.org/) is an open-source extensible mash-up platform supporting various data formats that are common on the Web, such as RDF, RDFa, XML, or microformats [4]. As a “Web Pipe” system, it allows one to aggregate data from distributed Web sources via SPARQL, XQUERY, and several other scripting languages, while preserving properties such as abstraction, encapsulation, component-orientation, code reusability, and maintainability. Semantic Web Pipes produce output streams of RDF data by filtering and transforming original RDF. Each published pipe or mash-up becomes itself a Semantic Web source that can be used in other mash-ups or by end-user applications. OntoPipeliner (ontoframe.kr/OntoPipeliner/main.html) is another semantic mash-up tool that allows the dynamic assembly of existing semantically operated services to help the user with designing and composing new services.

4.4 Related Resources

4.4.1 Books

A Semantic Web Primer (2nd Edition) by Grigoris Antoniou and Frank van Harmelen (ISBN-13: 978-0262012423, MIT Press): The primer provides a systematic approach to the different languages (e.g., XML, RDF, and OWL) and technologies (ontologies, logics, and inference) that are central to Semantic Web development. The second edition includes amongst many other extensions new material on SPARQL. Supplementary materials, including slides, online versions of many of the code fragments in the book, and links to further reading, can be found at www.semanticwebprimer.org.

Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL by Dean Allemang and James Hendler (ISBN-13: 978-0123735560, Morgan Kaufmann): This book offers insightful information to anyone who works with managing, sharing, and

accessing information. It provides a solid base of knowledge about the principles and technologies, as well as the main architectural building blocks of the Semantic Web. Additionally, it addresses more advanced topics such as inference, ontology languages, and various do's and don'ts of ontology engineering. In summary, the book is considered of interest to readers who wonder about the technicalities leveraging the future of data management on the Web.

Foundations of Semantic Web Technologies by Pascal Hitzler, Markus Krötzsch, and Sebastian Rudolph (ISBN-13: 978-1420090505, Chapman & Hall): This book focuses on ontology languages that are standardized or under standardization mostly by W3C: RDF Schema, OWL, Rules, and the corresponding query languages such as SPARQL. Interesting aspects of the book are the chapters about the very recent developments around OWL 2 and the Rule Interchange Format (RIF).

4.4.2 Softwares and Tools

Protégé (protege.stanford.edu): Protégé is a free, open-source Java ontology editor and knowledge-base framework. Its success is largely based on the extensible plug-and-play environment that makes it a flexible base for rapid prototyping and application development. Protégé supports RDF/S with appropriate import and export functionalities.

NeOn Toolkit (www.neon-toolkit.org): The NeOn Toolkit is a multi-platform ontology engineering environment. The toolkit is based on the Eclipse platform and provides plug-ins covering a variety of ontology engineering activities, such as ontology evaluation and matching, and reasoning and inference. The RDF Editor is a modeling tool for the creation and maintenance of semantic models in RDF. The NeOn Toolkit is covered in

► [Ontologies and the Semantic Web](#).

RDF2Go (rdf2go.semweb4j.org): RDF2Go is an abstraction over triple (and quad) stores that allows developers of semantic applications to program against rdf2go interfaces and choose or change the underlying repository implementation later easily. There are adapters available, amongst others, for Sesame, Jena, and OWLIM.

Sesame (www.openrdf.org): Sesame is an open-source RDF framework in Java for storage, RDFS inferencing, and querying via SeRQL and SPARQL. As part of Sesame, the RIO package offers a number of RDF parsers and serializers.

Jena (jena.sourceforge.net): Jena is a Semantic Web framework for Java. It provides a programmatic environment for RDF, RDFS, OWL, and a query engine for SPARQL (www.openjena.org/ARQ/); moreover, it includes a rule-based inference engine.

OWLIM (www.ontotext.com/owlim/): OWLIM provides a family of semantic repositories; that is, RDF database management systems. The OWLIM RDF engine is implemented in Java and fully compliant with Sesame. It has inference support for RDFS, OWL Horst, and OWL 2 RL. According to the claims of its developers, OWLIM is currently the most scalable RDF repository with the best performance figure in terms of loading and query evaluation.

Virtuoso (virtuoso.openlinksw.com): Virtuoso is an enterprise grade multi-model data server with support for various data formats, amongst many others RDF too. Virtuoso includes an RDF store, a SPARQL query engine with SQL integration, and various user front ends to access and manage semantic data. There is an open-source edition of Virtuoso often referred to as OpenLink Virtuoso (<http://sourceforge.net/projects/virtuoso/>).

AllegroGraph (<http://www.franz.com/agraph/allegrograph/>): The AllegroGraph RDF store offers a persistent RDF graph database with Enterprise Online Transaction Processing. AllegroGraph 4.0 supports SPARQL, RDFS++, and Prolog reasoning, and enhances this setting with temporal reasoning rules and a geospatial engine for complex event processing. There is a free server edition available that is limited to up to 50 million triples only. OWLIM, Virtuoso and AllegroGraph are covered in [▶ Storing the Semantic Web: Repositories](#).

Corese/KGRAM (<http://www-sop.inria.fr/edelweiss/software/corese/>) is a RDF engine based on Conceptual Graphs (CG). It enables the processing of RDFS, OWL-Lite, and RDF statements and it can perform SPARQL queries and run rules over the RDF graph. It relies on CG formalism to deeply exploit the graph nature of RDF and investigate graph-based extensions to the formalisms and the reasoning. This engine is developed in Java and focuses on providing an efficient stand-alone main-memory platform used in many research and development programs.

For a far more comprehensive listing of tools that are relevant to RDF, the interested reader is referred to the corresponding website of W3C: <http://www.w3.org/RDF/>, or the more linked data publishing related tool listing on <http://linkeddata.org/tools>.

4.4.3 Websites

<http://www.w3.org/RDF/>: This is the main page of W3C for all matters related to the Resource Description Framework. It refers to all relevant readings and lists various RDF tools and related standards.

www.w3.org/2001/sw: This is W3C Semantic Web Activity website and as such the main access point to all work carried out at W3C in regards to semantic technologies.

www.w3.org/TR/rdf-primer/: The RDF Primer by Frank Manola and Eric Miller is the starting point for getting to know more about RDF. It introduces its basic concepts and its XML syntax. It also explains how to define RDFS vocabularies and describes the content and purpose of the RDF specification documents.

www.linkeddata.org: The Linked Open Data community website provides a home for resources and information to promote and foster the linked data movement.

www.planetrdf.com: A collective website on RDF with aggregated entries of up to 50 Semantic Web-related Weblogs. Also available on the site at <http://planetrdf.com/guide/> is an extensive collection of documents, publications, discussions, and press releases from the earlier days of RDF (1998–2004).

4.4.4 Standards

RDF/S: <http://www.w3.org/standards/techs/rdf>

Concepts and Abstract Syntax: <http://www.w3.org/TR/rdf-concepts/>

Semantics: <http://www.w3.org/TR/rdf-mt/>

Primer: <http://www.w3.org/TR/rdf-primer/>

RDF Schema: <http://www.w3.org/TR/rdf-schema/>

RDF/XML Syntax: <http://www.w3.org/TR/rdf-syntax-grammar/>

4.5 Future Issues

Although RDF was published as a W3C recommendation back in 2004, and its specification has been stable since, there were several issues left open. These 21 issues were postponed for future versions of RDF (<http://www.w3.org/2000/03/rdf-tracking/>), and some of them were taken up in 2010 in a W3C workshop on the future of RDF. Several work items have now been identified for what should be considered in a next version of RDF. Two major evolutions to expect concern the model of RDF and the syntax, respectively.

The current RDF specification does not know the concept of graph, and the only official possibility to link metadata to some RDF statements is the cumbersome reification process. Although the reification of RDF triples is determined by the recommendation, many RDF practitioners clearly advise its usage. Still, graph identification and metadata on RDF graphs are widely recognized as being relevant for numerous use cases including the annotation of semantic data with provenance, authorship, creation date, and use-by date information, and with accuracy, authentication, certification, validity, access rights, copyrights, and many other informational context properties. The indication of such metadata becomes more and more important as the application of RDF for data publishing is increasing rapidly. Naturally, the use of so-called named graphs emerged without having ever received the status of a recommendation. Two particular application scenarios that are taking on importance at the time of writing, namely open government data and linked data mash-ups, call for a standard in this respect, in order to offer a common way to do provenance tracking, accuracy indications, and copyright information linking. Such metadata is crucial in managing trust relationships between data providers and consumers, and for ensuring proper usage of linked data in open and largely uncontrolled environments. The incorporation of these extensions to RDF requires both an evolution of the core RDF data model to include the notion of graph and a mechanism to name them, and also an evolution of the syntax to allow the naming.

The second major change that will happen in the context of RDF concerns its syntax. The principal syntax that is standardized and promoted by W3C is RDF/XML, which is both very verbose and hard to read for humans. Moreover, it is not possible to write down a graph (as of RDF) in a tree (as of XML) in an intuitive and obvious way. In the Semantic Web community, as also throughout this chapter, Turtle is a widely used syntax for RDF

that is unfortunately not standardized, and hence not largely adopted by people outside the community. Therefore, there is a strong demand from RDF experts to (even) better align Turtle with the SPARQL pattern syntax, to extend Turtle to support graph identification, and to eventually standardize it in order to ensure large-scale adoption and compatibility across implementations. In addition, although XML is still the predominant data format on the Web, more recently JSON (JavaScript Object Notation, www.json.org) has emerged as a lightweight interchange format for the Web. In fact, many services that are implemented on top of linked data encode the responses in JSON; not least as it is far easier for humans to read than XML. Consequently, first ideas are in development to provide an RDF/JSON serialization for the exchange of RDF graphs, and to eventually make this new format a standard specification too; see, for example, the Talix Wiki page on the RDF/JSON specification at http://n2.talis.com/wiki/RDF_JSON_Specification. This new serialization would provide Web developers (Javascript, HTML5, . . .) with an easier way to use RDF data within existing Web tools.

Further noteworthy items around RDF that are expected to be discussed at W3C concern the skolemization of blank nodes, the official use of IRIs instead of URIs for the identification of resources, and a rethinking of RDF semantics. In fact, although RDF carries an own semantics, it is mainly used as a data-structuring language only, for which RDF semantics has a minimal role.

A last issue to highlight is the one of RDF/S databases. Support for scalable database solutions is not a future issue, but in fact an ongoing one. Major database vendors like Oracle, IBM, or OpenLink are selling RDF-enabled solutions that build on their core relational database solutions. Other companies offer native RDF stores, or graph-based databases – some of them are listed under related resources – that support inference for RDFS or a more expressive language such as OWL. Still, even though support for RDF is increasing in mainstream database management systems, its adoption and benchmarking still lacks wider support, which hampers its integration in many promising industrial solutions, such as the earlier-mentioned networked economies, pharma industry, or telecoms and energy providers.

Complementary to the ongoing development of dedicated RDF repositories, there is work being carried out on bringing data from relational databases to the Semantic Web. Indeed, one of the main drivers of Semantic Web research has always been the exposure of vast amounts of relational database information on the Web, in such a way that it could be processed by machines. Development is ongoing along two major paths. Firstly, relational database content is mapped into RDF stores from where it can be queried by users as RDF; secondly, when working in real time, users query the relational database directly, and only the resulting bindings are mapped to RDF. At W3C there was a working group established that addresses exactly these mapping and integration problems (<http://www.w3.org/2001/sw/rdb2rdf/>). The objective of the working group is to standardize a language for mapping relational data and relational database schemas into RDF and OWL. Such a language enables the aggregation of relational data with other data on the Web, which further improves enterprise data integration. Essentially, enterprise data integration relies more and more on the exploitation of public data sources, in addition

to already heterogeneous company-internal data, in order to more adequately and quickly react to public opinion, changes of interest, or economic situations. In short, the RDB2RDF working group is in the process of establishing a standardized approach for an Extract-Transform-Load (ETL) procedure that transforms relational data into RDF. The working group is expected to conclude in the fall of 2011.

4.6 Summary

The Semantic Web is an extension to the Web in that it offers the necessary technologies to link data and share the semantics of their schemas. RDF evolved from various metadata standards and annotations formats to provide the first W3C recommendation on how to describe, publish, and link metadata on the Web. RDFS followed with the specification of language constructs to share the semantics of their schemas. RDF reuses the Web approach to identify resources (URI) and to allow the explicit representation of relationships between two or more resources. By using and reusing URIs, anyone can say anything about any topic, and anyone can add to it, thanks to the schema-last data model of RDF. The sum of all statements about a resource and the links to others yields the fundamental building block for the graph-based data structures of RDF that foster worldwide data integration. RDF Schema is a typing system to describe classes of RDF resources, their properties, and the relations between them. Classes and properties are viewed as sets, and RDFS relies on set inclusion, intersection, and union to define the semantics of these relations. RDF Schemas are written in RDF and inherit all of its technical advantages, such as the distributed and easily extensible data model, and the same query language. To this end, the combination of RDF and its schema language RDFS lays the ground for all the other activities of the Semantic Web and provides a fundamental building block for many other activities of W3C and increasingly in other bodies and industrial consortiums too.

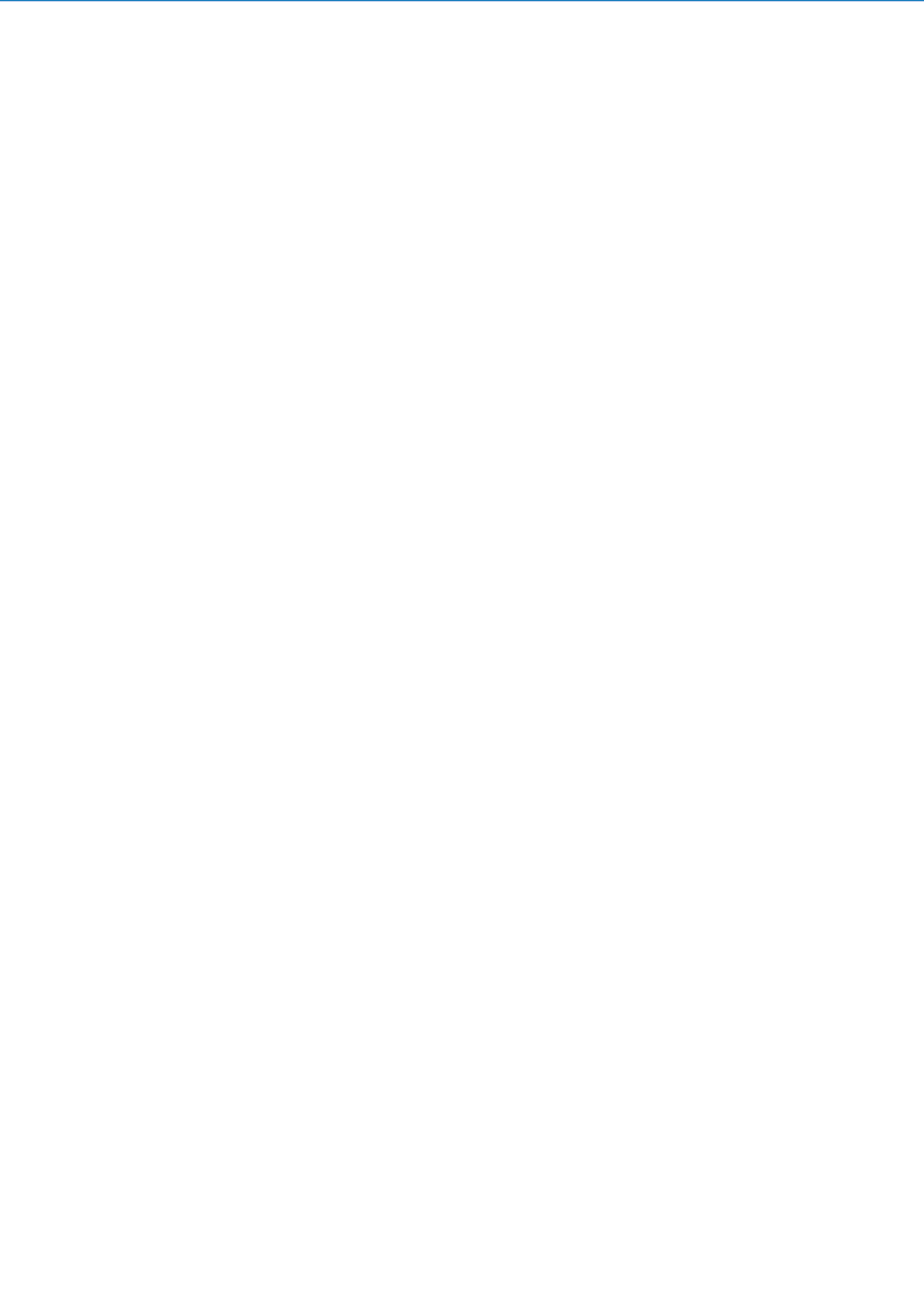
4.7 Cross-References

- KR and Reasoning on the Semantic Web: OWL
- Querying the Semantic Web: SPARQL
- Semantic Annotation and Retrieval: Web of Data

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. *Sci. Am.* **284**(5), 34–43 (2001)
2. Heath, T., Hausenblas, M., Bizer, Ch., Cyganiak, R., Hartig, O.: How to publish linked data on the web (tutorial). In: Proceedings of the Seventh International Semantic Web Conference (ISWC 2008), Karlsruhe. Lecture Notes in Computer Science, vol. 5318. Springer, Berlin. <http://events.linkeddata.org/iswc2008tutorial/how-to-publish-linkeddata-iswc2008-slides.pdf> (2008)

3. Hofstadter, D.R.: Gödel, Escher, Bach: An Eternal Golden Braid. Basic Books, New York (1979). ISBN 978-0465026562
4. Le Phuoc, D., Polleres, A., Morbidoni, Ch., Hauswirth, M., Tummarello, G.: Rapid semantic web mashup development through semantic web pipes. In: Proceedings of the 18th World Wide Web Conference (WWW 2009), Madrid (2009)
5. Patel-Schneider, P.F.: Building the semantic web tower from RDF straw. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005), Edinburgh, pp. 546–551 (2005)
6. Patel-Schneider P.F., Fensel D.: Layering the semantic web: problems and directions. In: Proceedings of the First International Semantic Web Conference (ISWC 2002), Sardinia. Lecture Notes in Computer Science, vol. 2342, pp. 16–29. Springer, Berlin (2002)



5 Semantic Annotation and Retrieval: Web of Hypertext – RDFa and Microformats

Ben Adida^{1,4} · Mark Birbeck² · Ivan Herman³

¹Creative Commons, San Francisco, USA

²WebBackplane, London, UK

³World Wide Web Consortium and CWI, Amsterdam, The Netherlands

⁴Harvard University, Cambridge, MA, USA

5.1	<i>Scientific and Technical Overview</i>	159
5.1.1	Introduction	159
5.1.1.1	Applicability of Technologies	161
5.1.2	Microformats and the Semantic Web	162
5.1.2.1	Microformats	162
5.1.2.2	Transforming Microformats to RDF	165
5.1.2.3	GRDDL	166
5.1.3	RDFa	168
5.1.3.1	Adding Triples with Literals	169
5.1.3.2	Nonliteral Objects	172
5.1.3.3	Specifying Subjects	173
5.1.3.4	Grouping Objects	175
5.1.3.5	Typing	176
5.1.3.6	Local (Blank) Nodes	177
5.1.4	GRDDL and RDFa	178
5.2	<i>Example Applications</i>	179
5.2.1	Connections Between the Human and Machine-Readable Web	179
5.2.2	Search	180
5.2.2.1	Yahoo! SearchMonkey and BOSS	180
5.2.2.2	Google	181
5.2.3	Facebook's Open Graph Protocol	181
5.2.4	Browser Extensions	182

5.2.4.1	BlueOrganizer	182
5.2.4.2	Operator	183
5.2.4.3	Fuzz	183
5.2.5	Creative Commons and the CC Network	183
5.2.6	Separating User Interface from Raw Content	184
5.2.7	Decentralized Data Distribution Using UK Government Websites	184
5.2.8	Publication of Vocabularies and Datasets	185
5.2.9	Self-documenting Vocabulary Specification	187
5.3	<i>Related Resources</i>	188
5.4	<i>Future Issues</i>	188
5.5	<i>Cross-References</i>	189

Abstract: For the Semantic Web to succeed in a major public setting, it needs to leverage the existing Web. Two major technologies have emerged over the last few years to bridge the vast, existing “clickable” Web and the new Semantic Web: Microformats and RDFa. Both allow authors to embed extra information within (X)HTML to mark up the structure, not just the visual presentation, of the information they publish. In this chapter, both approaches are explained, exploring their strengths and weaknesses, providing example applications, and touching on future considerations.

5.1 Scientific and Technical Overview

5.1.1 Introduction

Some of the introductory ideas of both the Semantic Web and RDF are presented elsewhere in this book. But a key question is how exactly might an existing Web publisher become part of the Semantic Web – in short, where does one start?

There are indeed a wide range of technologies and software packages that allow a number of sophisticated things to be done with semantic information. Ever since the start of the Semantic Web, one of the issues has been the availability of various types of data for further integration. Technically, this means making data available in RDF. One approach is to encode the RDF data in one of its serialization formats, like RDF/XML [1] or Turtle [2], but that approach requires publishers to make datasets available in a specific format whose sole purpose is the Semantic Web. Instead, interfaces to databases are being developed that can, for example, provide on-the-fly conversion of existing data into RDF, often via SPARQL [3] end points. Automatic or semiautomatic conversions exist for a number of other formats. In general it has been recognized that one should not look for one specific approach only; rather, different types of data on the Web require their own, data-specific way of expressing their content in RDF.

One of the obvious and major sources of data on the Web is, of course, hypertext, that is, HTML and XHTML. HTML is not only used by individual authors to produce Web pages directly; in fact, most of the HTML pages visible through browsers are produced by various types of back-end processes. The information itself, displayed through HTML, is often just a reflection of database content (often with significant user-interface embellishment). HTML is, therefore, a natural vehicle to incorporate the data that, ultimately, can be processed by the Semantic Web. This chapter will concentrate on how to piggyback Semantic Web data within existing hypertext, so that it can be consumed by generic Semantic Web Services.

Conveniently, the (traditional) Web community has, in parallel, come to the realization that it is beneficial to reveal more of the underlying structured information than what can simply be displayed by a browser for, essentially, human consumption. The Web 2.0 phenomenon brought a more interactive and participative Web to the fore that often relies on additional structure within the published hypertext data. Take, for example, a typical mash-up site like TripIt (<http://tripit.com>), which relies on the availability of data

regarding flights, weather information, or online maps. At the moment, getting to such data in an automated fashion is complicated and often involves scraping, that is, an ad hoc interpretation of various websites, or accessing site-specific APIs. Obviously, this approach does not scale to the entire Web very efficiently beyond a small handful of source websites.

This dual evolution (on the Semantic and on the more “traditional” Web) saw the emergence of a common approach (though with different techniques), namely that of (X)HTML pages including additional information that provide structural, “meta” information about what is being displayed visually. This additional information is added to the (X)HTML content as elements or attributes, providing additional information to the text. For example, if an HTML element contains the words “Ivan Herman,” then an extra attribute can denote the fact that this is the name of a human being. If that extra attribute is based on some general conventions, then automated processors, reading that (X)HTML file and analyzing its structure, can automatically infer that the string is indeed a person’s name and process it accordingly.

The traditional Web community has developed the Microformats approach [4] for that purpose. Microformats reuse existing HTML attributes, for example, `class` and `title`, and, sometimes, elements (e.g., `abbr`). In doing so, they ensure a quick and easy deployment of such annotated HTML files through existing editors and other tools. Microformat vocabularies are defined and maintained by the community; vocabularies (i.e., conventions on which elements and attributes are used and for what purpose) cover different application areas. Applications that understand a specific microformat vocabulary and know that a specific (X)HTML page abides to that vocabulary can make use of this inherent structure in the markup to get at the raw data without screen-scraping.

Although Microformats have not been developed with the Semantic Web as their target, technologies exist to convert, if necessary, XHTML files with microformat content into RDF. The transformation engine itself is, typically, XSLT [5] that converts the XHTML structure into RDF/XML (note that different XSLT scripts must be available for different Microformat vocabularies). A W3C standard, called GRDDL [6], has been defined to enable generic processors to locate the XSLT transformation for a particular Microformat, given an appropriate declaration in the XHTML content header.

Microformats provide an easy solution for simple data structures such as calendar events and contact information. However, beyond these simple cases, Microformats may become difficult to scale. The centralized Microformat definition process, which is required to ensure that one Microformat does not interfere with another (given that Microformat syntax may vary from one vocabulary to the next), significantly slows the adoption of new vocabularies, and prevents the definition of more complex vocabularies altogether. The centralized process is also required to prevent vocabulary clash: if the same HTML content contains microformat data for two different vocabularies, and both happen to use, say, the `title` attribute, how would one decide on the intended meaning of that attribute? Finally, there is also an additional clash between the usage of the HTML attributes for Microformat purposes and their intended usage in HTML: Microformats, effectively, “hijack” certain attributes for purposes that are semantically different from their original design. As a result, Microformats may clash with existing standards or usage

patterns; for example, the `abbr` element, combined with a `@title` attribute, is often used in assistive technologies.

RDFa [7], defined by the World Wide Web Consortium (W3C), provides a more general, albeit more complex approach. Just like Microformats, RDFa relies on attributes; however, RDFa defines its own set of attributes that are used alongside existing XHTML constructs. (In contrast to some Microformat dialect, RDFa does not rely on, or introduces novel XHTML elements.) Furthermore, the values of these attributes are firmly rooted in the usage of URIs, a unique way of identifying each term; vocabulary clashes are therefore automatically avoided. This way, RDFa provides the flexibility to add essentially any information to an XHTML page, provided that a suitable vocabulary is defined that gives sense to the URIs employed by the user. All these features are, obviously, closely related to RDF and are based on the same principles. Indeed, the RDFa specification defines a precise mapping from an XHTML+RDFa file to RDF. In contrast to the Microformat+GRDDL approach, there is no need to define a separate transformation engine per vocabulary; instead, one such engine can cover all usages of XHTML+RDFa, hence the suitability of RDFa to wide-scale usage. In fact, RDFa+XHTML may be viewed as another RDF serialization format, alongside RDF/XML or Turtle, one that provides both human and machine readability. (Note, however, that RDFa does not provide syntactic shorthands for some RDF constructions like containers and collections.)

Both approaches use the structure of an (X)HTML document as a scaffold on which to attach RDF information. By using HTML in this way, RDF can be carried along wherever HTML “goes.” In other words, publishing semantic information is now as easy as setting up a blog on Blogspot (<http://blogspot.com>), or using a content management system like Drupal. For advanced content mapped to a database, RDFa can be produced by a Rails, Struts, or ASP application on the server. In short, anyone looking to publish semantic information can use the techniques that they are already using for publishing (X)HTML.

As mentioned previously, there are already many Web applications which read data from other Web pages by examining the markup and using various heuristics to extract where the content may be. This process is often called *screen-scraping* and typically involves some stable knowledge of the target page layout, so that the proper information can be plucked from the expected position in the DOM tree. However, if the layout changes significantly, all bets are off, and the screen-scrafer will need to be reconfigured. With techniques like Microformats+GRDDL or RDFa, with the RDF information embedded into the web-pages, there is no need for screen-scraping, since the position and type of the information is made explicit. Hence the importance of the techniques is described in this chapter.

5.1.1.1 Applicability of Technologies

In what follows, GRDDL and RDFa will be presented only in conjunction with (X)HTML. It must be noted, however, that those technologies can also be used for other XML dialects. GRDDL has been defined in such a way that a generic processor could find a suitable XSLT transformation for any XML schema or namespaces. Similarly, the RDFa attribute set can

be added to other XML dialects; this has already been done for SVG Tiny 1.2 [9] or Yahoo!’s MediaRSS [10].

5.1.2 Microformats and the Semantic Web

(See also ► [Table 5.1](#) for an overview on Microformats and GRDDL.)

5.1.2.1 Microformats

Microformats reuse existing attributes in the HTML specification to add semantic information to the content. The term “Microformats” does not refer to one specific vocabulary, but rather to a general approach of using these attributes. Each application is assigned its own vocabulary and associated syntax. For example, hCard is used to describe people and organizations, hCalendar to publish calendar data, and hAtom for news feeds. The community around the <http://www.microformats.org> site is set up to shepherd the process of developing and standardizing new Microformats. These Microformat vocabularies are rarely defined from scratch; instead, they are a translation of existing vocabularies to the Microformats approach (e.g., hCalendar is a translation of vCalendar, hAtom of ATOM, hCard for vCard, etc.).

Technically, the semantic information itself is encoded in attribute values using simple strings. The definition of a new Microformat consists of defining these strings and giving a textual description as to their intended meaning. For example, the hCalendar specification [10] specifies the meaning of “dtstart” (the starting date of a calendar event), “location” (the location of a calendar event), or “geo” (the latitude and longitude values of the location). The attribute of choice to encode these values is the @class attribute.

As a very simple example, the following HTML:

```
<div class="vevent">
  <a href="http://2008.sxsw.com/interactive/" class="url">
    <span class="summary">SXSW Interactive</span>
  </a>
</div>
```

■ **Table 5.1**

Common problems and their solutions using Microformats + GRDDL

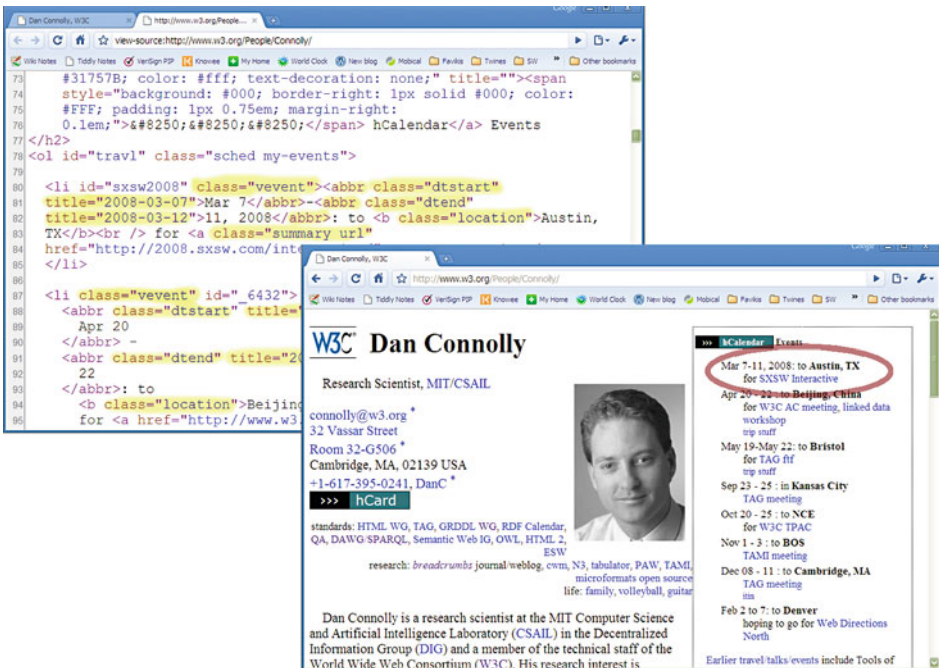
Problem	Solution in Microformat + GRDDL
Define terms in Microformats	Define attribute values, usually for @class, (sometimes for @title or the @abbr element)
Convert terms to RDF	Apply a GRDDL transformation (usually XSLT)
Find GRDDL transform (general)	Use @profile with generic value and a <link> element with @rel="transformation"
Find GRDDL transform (for specific, frequent cases)	Use @profile with a specific value (leading to another document to be GRDDL-d)

provides the information that (1) this is an event, (2) it can be summarized as “SXSW Interactive,” and (3) it has a homepage (the usual approach is to use the @href attribute value as the URI corresponding to the “url” tag).

It is, in some cases, not possible to rely exclusively on @class and the existing HTML content structure; extra elements and possibly other attributes should also be used. A typical example, in hCalendar, is the setting of dates, which usually relies on the abbr element and uses the @title attribute for the precise (i.e., ISO formatted) date value instead of the human-readable form. So, for example, the code above could be extended by adding the start and end dates of the event as follows:

```
<div class="vevent">
  <abbr class="dtstart" title="2008-03-07">Mar 7</abbr>-
  <abbr class="dtend" title="2008-03-12">11, 2008</abbr>:
  <a href="http://2008.sxsw.com/interactive/" class="url">
    <span class="summary">SXSW Interactive</span>
  </a>
</div>
```

► [Figure 5.1](#) is a typical example for using Microformats. The page uses two different Microformats (hCard and hCalendar) in an otherwise typical Web page, denoting the



■ Fig. 5.1

Typical example of hCalendar usage

address data and conference events of the person. The extra information is embedded in the XHTML page but those extra attributes are not visible on the browser screen. The page reads just like any other page. The figure also shows what is behind the screen, with some of the hCalendar tags highlighted. Tools exist to extract, for example, iCalendar data from a hCalendar Microformat markup, yielding (for this example):

```
BEGIN:VCALENDAR
...
BEGIN:VEVENT
URL:http://2008.sxsw.com/interactive/
LOCATION;CHARSET=UTF-8:Austin\, TX
SUMMARY;CHARSET=UTF-8:SXSW Interactive
DTSTART:20080307
DTEND:20080312
END:VEVENT
```

Microformats are usually registered on the community site [4], after discussions on dedicated mailing lists, blogs, or IRC channels. In this sense, it is a very typical case of a “grassroots” effort. The site repertories Microformat specifications, both in final and draft formats. These include the hCalendar and hCard specifications used in the examples, but also Microformats for audio content, simple voting procedures, or for publishing resumes and CVs.

Microformats provide a quick and effective way to add a little bit of semantics to HTML pages. The fact that they reuse existing HTML attributes and elements also means that it is fairly easy to edit HTML pages with Microformats (using various HTML editors) and adding the microformat attribute values does not endanger the validity of the page with the usual HTML DTDs and validators. The relative simplicity of the terms makes it also easy to deploy Microformats.

Microformats have some drawbacks, however. The main issue is that the semantic meaning described in the microformat tags are all defined in isolation, independently of one another. Even if the same HTML file contains different Microformats (as in the example on [Fig. 5.1](#)), developing an application that integrates the data from those two tag-sets means a case-by-case development in understanding what all the different tags stand for, how they are encoded, what parsing procedures should be followed, etc. This integration task may be made more complicated by a possible clash in the microformat specifications; what happens if two different Microformat dialects use the same attribute value or the same HTML element for different purposes? Typically, the Microformats community ensures that no conflicts between different Microformats arise, resulting in sometimes confusing semantic choices for simple terms, for example, “title” means “Job Title” because hResume was the first Microformat to lay claim to it. However, the centralized control to prevent conflicts means that either no decentralized innovation will occur, or name collisions will.

5.1.2.2 Transforming Microformats to RDF

Further complications arise with the usage of Microformats if the encoded semantic meaning is to be integrated with different sources of data that are not necessarily available in an HTML file. This integration aspect is the central paradigm of the Semantic Web, with RDF [11] playing the role of a glue among different data schemas. As a consequence, a natural way of integrating Microformats into the Semantic Web – while still keeping their benefits – is to extract the semantic content and convert it into RDF.

If the host language is XHTML, that is, a dialect of XML, then the straightforward way of doing that is to use a suitable XSLT [5] transformation. XSLT is a functional style programming language, itself encoded in XML, which describes the transformation of an XML tree into either another XML tree or simply into plain text. Since its inception, XSLT has become an integral part of the XML infrastructure. Implementations of XSLT processors are widely available as stand-alone programs and as part of XML editors or Web browsers. Due to the simplicity of the usual Microformat dialects, the definition of an XSLT transformation to extract the semantic data and generate an RDF equivalent in RDF/XML (or Turtle) is usually a relatively straightforward task. Here is an example of the RDF output an XSLT processor may produce for the example above:

```
<http://www.w3.org/People/Connolly/#sxsw2008>
  a<http://www.w3.org/2002/12/cal/icaltzd#Vevent>;
  <http://www.w3.org/2002/12/cal/icaltzd#summary>
    "SXSW Interactive";
  <http://www.w3.org/2002/12/cal/icaltzd#dtstart>
    "2008-03-07"^^xsd:date;
  <http://www.w3.org/2002/12/cal/icaltzd#dtend>
    "2008-03-12"^^xsd:date;
  <http://www.w3.org/2002/12/cal/icaltzd#url>
    <http://2008.sxsw.com/interactive/>;
  <http://www.w3.org/2002/12/cal/icaltzd#location>
    "Austin, TX".
```

The transformation depends on an RDF vocabulary that is equivalent to the Microformat dialect (the `.../icaltzd` namespace in this example). Thus, to transform the Microformat data into RDF, thereby integrating with other types of data on the Web, two steps have to be followed:

1. Define an RDF vocabulary corresponding to the Microformat dialect.
2. Create an XSLT transformation that extracts the data from the XHTML source and produces RDF using that vocabulary.

A number of XSLT transformations have already been defined and published for the more widely used Microformats. A wiki site has been set up at the W3C [12] listing some of these, including references to the RDF vocabularies as well as to the XSLT transformations

themselves. The site is maintained by the community at large, and is not under the official control of any W3C or other groups.

5.1.2.3 GRDDL

The XSLT transformation approach described in the previous section does not cover all what is needed to smoothly integrate Microformats with the Semantic Web. Indeed, even though transformations may be defined, the issue of finding the appropriate transformation remains. When a processor finds an XHTML page containing a Microformat, how does that processor know which transformation to apply? By default, the Microformat dialect itself does not provide an answer.

This is where an additional W3C technology, called GRDDL [6], comes in. GRDDL uses existing (X)HTML constructs to locate the necessary transformation(s) for the HTML markup. A generic GRDDL processor would then find and download the transformation (at least conceptually, because an optimized GRDDL processor would probably cache the transformation for all major Microformats), execute the transformation, and produce the RDF output. This means that fully generic and automatic GRDDL processors can be created and added to general RDF application environments. Put another way, a user can point at a URI of a GRDDL-d page annotated with Microformats just as if it was a URI referring to an RDF/XML file.

What are the additional attributes defined by GRDDL? The simplest way of defining the GRDDL transformation is to modify the header of an XHTML file as follows:

- Add a `@profile` attribute to the XHTML head denoting the fact that the XHTML file is to be processed by a GRDDL processor; and use a “link” element in the head to indicate the URI of the transformation.

For example, the previous example involving the hCalendar Microformat could have the following header:

```
<html>
  <head profile="http://www.w3.org/2003/g/data-view">
    <link rel="transformation"
      href="http://www.w3.org/2002/12/cal/glean-hcal.xsl"/>
    ...
  </head>
  ...
</html>
```

If the page uses several Microformats, separate link elements should be added for each of those. The GRDDL processor would then execute each transformation separately and perform an RDF merge on the generated graphs (as described in the RDF Semantics document [11]), yielding the final result.

GRDDL also defines a way to add the necessary attributes to a dialect of XHTML files, thereby simplifying the task of authors. Whereas the profile value <http://www.w3.org/2003/g/data-view> simply means “process this file through a GRDDL processor,” publishers can also use dialect-specific `@profile` values. The goal of these specific values is to “merge” the two items of information above, that is, making the separate link element unnecessary.

Technically, what happens is as follows:

1. The GRDDL processor follows this special profile URI, retrieves the target (i.e., profile) document.
2. Performs a GRDDL transform on the profile document, yielding a (small) RDF graph.
3. This RDF graph contains a reference to another transformation.
4. This transformation is applied on the original source.

One could say that the usage of the dialect-specific profile leads to some sort of an indirect application of GRDDL.

For example, the file at the URI <http://www.w3.org/2002/12/cal/hcal> contains the necessary GRDDL information to yield the RDF triple (the predicate URI is defined by the GRDDL standard):

```
<> <http://www.w3.org/2003/g/data-view#profileTransformation>
    <http://www.w3.org/2002/12/cal/glean-hcal.xsl>.
```

The object of this triple is identical to the transformation provided by the simple GRDDL markup for hCalendar. The XHTML source of the example can therefore be modified to (note the removal of the link element):

```
<html>
  <head profile="http://www.w3.org/2002/12/cal/hcal">
  </head>
  ...
</html>
```

Through the indirect step, the GRDDL processor will produce the same RDF graph as before.

As mentioned above, many RDF environments, for example, Redland (<http://librdf.org>) or Jena (<http://jena.sourceforge.net/>), or RDF browsers like the Tabulator (<http://www.w3.org/2005/ajar/tab>) already include automatic GRDDL processors in their newer releases. What this means in practice is that XHTML files with Microformats and with the necessary GRDDL attribute values will be processed, and the resulting RDF graph(s) added to, for example, a SPARQL query dataset automatically.

As already referred to in the introduction, the attributes defined by the GRDDL recommendation have their purely XML variants that do not depend on the XHTML structure (i.e., the `html` and `link` elements). The indirect approach is then achieved through the namespace document instead of the profile document. This turns GRDDL into an extremely powerful technology to build a bridge between the XML world and the

RDF world in general. The interested reader should refer to the GRDDL Recommendation [6] or the GRDDL Primer [13] for further details.

5.1.3 RDFa

RDFa shares a number of high-level design principles with Microformats. Both use (X)HTML attributes to express the additional data that can be retrieved from the (X)HTML content and converted into RDF, and both rely on external processing to retrieve the necessary information by interpreting the structure of the document. RDFa is, however, more flexible: if an RDF vocabulary has been defined by a community for some purpose, that vocabulary can be directly reused by RDFa publishers, whereas Microformats must redefine that vocabulary, through some additional community agreement, into a microformat dialect.

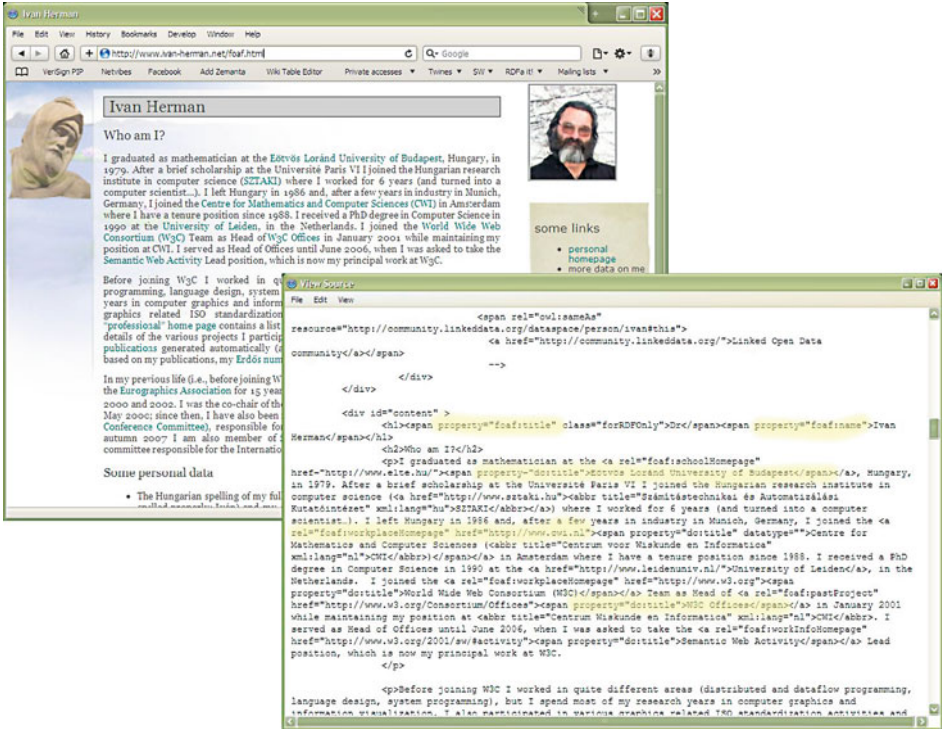
As an example (used later in this section), a widely used vocabulary in RDF applications is based on FOAF [14] (Friend-of-a-Friend). FOAF is used to describe personal data (names, phone numbers, etc.) as well as the relationships between people (e.g., that one person knows another). By publishing FOAF files about themselves, people can export, for example, their social relationships independently of a particular social network application. In practice, personal profile files use a number of different vocabularies beyond FOAF; indeed, the FOAF vocabulary itself tries to be minimalistic and relies on the usage of other terminologies to express, for example, geographical coordinates or contact addresses. Mixing various terminologies in the FOAF context is no problem in RDF – after all, integrating various types of vocabularies is one of the main advantages of using it.

FOAF files can be created by various tools or indeed created directly in, say, a text editor using an RDF serialization like Turtle. However, one has to realize that the data put into a FOAF file are almost identical to what one would put into one's CV: name, surname, schools, personal interests, e-mail addresses, etc. It is therefore a fairly obvious wish to merge these two; one should write a single XHTML file to publish a CV, and RDF applications should be able to extract the FOAF information automatically for further integration with other types of data.

Using Microformats to achieve this does not really work. First of all, the FOAF vocabulary is relatively large, which means that a hypothetical Microformat version, as well as a necessary GRDDL/XSLT transform, would become fairly complex. But, and perhaps more importantly, the fact that FOAF files typically mix many different vocabularies would make it unmanageable, if not outright impossible, for Microformats to produce CVs in XHTML that could also be used to automatically generate FOAF data (🔗 Fig. 5.2).

All this is not a problem for RDFa. Indeed, RDFa is perfectly well prepared to handle several vocabularies within one file. This is because RDFa is based on the usage of URIs, meaning that usage of various terms can be interleaved without any particular problems. Because vocabularies defined on the Semantic Web are also based on URIs, this means that RDFa can readily incorporate and benefit of the existence of many vocabularies and ontologies that the Semantic Web community develops, regardless of the complexity of these.

To make the usage of URIs easier, RDFa employs a namespace-like mechanism to abbreviate URIs via strings like `dc:title` or `foaf:Person` to replace full URIs (those



■ Fig. 5.2

CV file in XHTML with RDFa markup

URIs can be confusingly long). Users can declare these prefixes using the `xmlns:` attributes on any element in the XHTML hierarchy. These “compact URIs,” or CURIEs, are used in specific attributes, such as `@rel` and `@property`. When an attribute needs to use either a URI or a CURIE, for example in `@about`, CURIEs are differentiated using square brackets, for example, `[prefix: suffix]`. Those cases are, however, rare.

5.1.3.1 Adding Triples with Literals

So how does RDFa work? This section will use the example of FOAF files to introduce the various RDFa constructs. The specification of RDFa is defined in terms of RDF; essentially, what it does is to define what kind of RDF triples a specific combination of XHTML elements and attributes yield (noting that the attributes may be RDFa specific or, in some cases, native XHTML ones). This section will follow the same approach. (See also [Table 5.2](#) for an overview of the RDFa constructs presented here.)

One obvious entry in a CV file is one’s name. The XHTML file may look something like:

```
<h1>Ivan Herman</h1>
```

An RDFa attribute can be added to this statement:

■ **Table 5.2**

Common problems and their solutions using RDFa

Problem	Solution in RDFa
Abbreviate long URIs	@xmlns and CURIEs
Define a predicate with a literal object	@profile and the text content of the element
Define a predicate with a literal object (alternative)	@profile and @content
Set language tag of a literal	Qxml:lang
Set datatype of a literal	@datatype
Define a predicate with a URI resource object	@rel
Define a URI resource object	@href or @resource
Define the subject	@about (or inherit the value of @resource from the hierarchy)
Set type of a resource (rdf:type)	typeof
Define a blank node	_:xx style CURIE, or @typeof without a @resource or @href on the element

```
<h1 property="foaf:name">Ivan Herman</h1>
```

The extra @property instructs an RDFa-aware tool that whatever is enclosed in an element should be taken verbatim as literal, that is, to generate the following triple:

```
<> foaf:name "Ivan Herman".
```

Both in the XHTML/RDFa encoding and the final triple the term foaf:name stands for the abbreviation of a URI of the form:

```
http://xmlns.com/foaf/0.1/name
```

The pairing of the foaf prefix and its corresponding URI is done, in the case of RDFa, using an attribute declaration of the form:

```
xmlns:foaf="http://xmlns.com/foaf/0.1/"
```

somewhere in the XHTML hierarchy. (Sometimes, for convenience in templating systems, it will be placed at the top, that is, on the html element, but it can also be elsewhere.)

The example above defines a triple with a literal object, where the literal is automatically drawn from the XHTML text. Such a literal, in some cases, might be in another language than English. For example, the same file can also include the following:

```
The Hungarian spelling of my full name is
```

```
<span property="foaf:name" xml:lang="hu">Herman Iván</span>
```

that yields the following triple:

```
<> foaf:name "Herman Iván"@hu.
```

which is the Turtle notation to denote an RDF literal with a language tag. Note that RDFa does not introduce a separate attribute to denote a language; instead, the standard XHTML attribute `xml:lang` is reused. (The current version of RDFa does not use the `@lang` attribute, because that was not used by XHTML1.1 at the time of the specification. That has recently changed, so future versions of RDFa will also allow the usage `@lang`.)

Literals can also have datatypes, like floats, dates, integers, etc. For example, one could write:

```
<span property="foaf:birthday"
  datatype="xsd:date">1955-02-24</span>
```

leading to:

```
<> foaf:birthday "1955-02-24"^^xsd:date.
```

using, again, the Turtle notation to denote a triple with an RDF Literal object with a datatype. RDFa is silent on what kind of datatypes can be used, the value of `@datatype` is simply a (compact) URI. In practice, in accordance with the vast majority of RDF applications, the XSD datatypes [15] are used.

The last example also shows that, in some cases, the approach of relying on the XHTML text for literals is not optimal. Indeed, though the term `1955-02-24` is the standard way of denoting a date, it does look slightly unnatural in an English text. RDFa, therefore, introduces the `@content` attribute that can be used to denote a literal that is not part of the XHTML text. The example above could have been written as:

```
<span property="foaf:birthday" content="1955-02-24"
  datatype="xsd:date">24th February, 1955</span>
```

yielding exactly the same RDF triple but making the XHTML text more readable.

Speaking of dates, it is usually a good practice to add a date to the page, signaling the last update of the CV (as well as the generated FOAF data). This can easily be added to the page, using:

```
<span property="dc:date" datatype="xsd:date">2009-03-30</span>
```

The only difference is the usage of a different vocabulary, in this case a so-called Dublin Core [16] term. Adding this to the RDFa file is easy; the only requirement is to add another `xmlns:` attribute somewhere defining the URI for the `dc:` prefix; from that point on the FOAF and Dublin Core properties can be intermixed within the same RDFa file, and the RDFa processor would automatically generate the right triples. The same mechanism can be used with any number of vocabularies making it possible to mix several vocabularies within one XHTML/RDFa file easily.

`@property` attributes can list several values, in case triples with identical subjects and objects but with different predicates are to be generated. For example, it is possible to say:

```
<span property="dc:date dc:created"
  datatype="xsd:date">2009-03-30</span>
```


yielding two triples instead of one:

```
<> dc:date "2009-03-30"^^xsd:date.
<> dc:created "2009-03-30"^^xsd:date.
```

5.1.3.2 Nonliteral Objects

Of course, not all objects in triples are literals. For example, the FOAF vocabulary includes a term called `foaf:workplaceHomepage`. The object of a triple using this predicate should be a URI resource rather than a Literal. To differentiate between the two types of possible objects, RDFa uses the existing `@rel` and `@href` attributes for nonliteral objects. Using these the code

```
I joined the <a rel="foaf:workplaceHomepage"
  href=http://www.w3.org> Wide Web Consortium (W3C) </a>
```

yields

```
<> foaf:workplaceHomepage <http://www.w3.org>.
```

The `@rel` attribute is not specific to RDFa; it is a valid XHTML attribute although, in traditional XHTML, its usage is restricted to the `link` and `a` elements. RDFa extends its usage to any element. (Conceptually, the RDFa usage of `@rel` is compatible with the original interpretation in XHTML, although expressed in RDF terms.)

In some cases, especially when a clickable anchor element is used, the intended RDF object is not exactly the same as the clickable link. This is similar to the case where `@content` is used to “override” the human-readable data for machine-readability purposes. When it comes to URI objects, the method for such an override is to use `@resource` instead of `@href`. When both are used, only `@resource` is considered for RDFa purposes. Thus, it becomes possible, when a resource has two different URIs for machine and human readability, to link to both, one for each audience.

It is also possible to use both the `@rel` and the `@property` attributes on the same element; since their interpretation does not collide, it may be a succinct way of expressing several triples, one that considers the link target as a URI object, and another that considers the anchor text as a literal object. For example, something like

```
<span property="dc:title" rel="foaf:workplaceHomepage"
  resource="http://www.w3.org">
  World Wide Web Consortium (W3C) </span>
```

yields

```
<> foaf:workplaceHomepage <http://www.w3.org>.
<> dc:title "World Wide Web Consortium (W3C)".
```

Note that this version of the code uses the `span` element and the `@resource` attribute, meaning that the element is not clickable on the browser screen. The same triple is generated nevertheless.

5.1.3.3 Specifying Subjects

Up until this point all generated triples used `<>` as subjects, denoting the resource with the base URI. While this is fine for simple cases, it may not work for more complex examples. To address this issue, RDFa introduces the `@about` attribute. This attribute (reminiscent of `@rdf:about` in RDF/XML) makes a subject explicit. Specifically, the very first example would be, more realistically:

```
<h1 about="http://www.ivan-herman.net/foaf#me"
  property="foaf:name">Ivan Herman</h1>
```

generating:

```
<http://www.ivan-herman.net/foaf#me> foaf:name "Ivan Herman" .
```

An important aspect of `@about` is that it does not only specify the subject for the element where it appears, but for all descendants, too (unless it is overwritten by another `@about`). In other words, the following, slightly more complex RDFa code:

```
<div about="http://www.ivan-herman.net/foaf#me">
  <h1 property="foaf:name">Ivan Herman</h1>
  <ul>
    <li><a rel="foaf:homepage"
      href="http://www.ivan-herman.net">personal homepage</a></li>
    <li><a rel="foaf:weblog"
      href="http://www.ivan-herman.name">personal blog</a></li>
    ...
```

generates the following triples:

```
<http://www.ivan-herman.net/foaf#me> foaf:name
  "Ivan Herman" .
<http://www.ivan-herman.net/foaf#me> foaf:homepage
  <http://www.ivan-herman.net> .
<http://www.ivan-herman.net/foaf#me> foaf:weblog
  <http://www.ivan-herman.name> .
```

In other words, using `@about` makes it possible to group a number of triples sharing the same subject. Note that this is not unlike the Turtle idiom of regrouping

triples using the ‘;’ separator; indeed, the previous Turtle triples could have been written as:

```
<http://www.ivan-herman.net/foaf#me>
  foaf:name "Ivan Herman" ;
  foaf:homepage <http://www.ivan-herman.net> ;
  foaf:weblog <http://www.ivan-herman.name>.
```

which does show similarities to the structure of the XHTML code. (RDF/XML has similar simplification possibilities.)

Using @about is not the only way to set the subject. Another approach is to make use of nesting in RDFa. Nesting means that a @href (or @resource) attribute not only sets the object of a triple but also acts as an implicit @about for all descendant nodes. The easiest way to understand that is again in an example. Consider the following:

```
<div about="http://www.ivan-herman.net/foaf#me">
...
  I joined the <a rel="foaf:workplaceHomepage"
    href="http://www.w3.org">Wide Web Consortium (W3C)</a>
...

```

which yields, as before:

```
<http://www.ivan-herman.net/foaf#me>
  foaf:workplaceHomage <http://www.w3.org>.
```

However, one may want to add some additional triples with <http://www.w3.org> as subjects, for example, using the (Dublin Core) dc:title predicate to denote the title of a resource. The nesting feature of RDFa makes it possible as follows:

```
<div about="http://www.ivan-herman.net/foaf#me">
...
  I joined the <a rel="foaf:workplaceHomepage"
    href="http://www.w3.org">
    <span property="dc:title">World Wide Web Consortium (W3C)
    </span> </a>
...

```

The span element is nested as a child element of a; using the nesting rule of RDFa means that the following triples will be generated:

```
<http://www.ivan-herman.net/foaf#me>
  foaf:workplaceHomage <http://www.w3.org> .
<http://www.w3.org>
  dc:title "World Wide Web Consortium (W3C)" .
```

5.1.3.4 Grouping Objects

As already noted, @about can be used to group triples that share the same subject. Another frequent situation in practice is when a number of triples share not only the subject but the subject and the predicate. Consider the following situation in RDFa:

```
<div about="http://www.ivan-herman.net/foaf#me">
...
  <ul>
    <li>
      <a rel="foaf:holdsAccount"
        href="http://www.dopplr.com/traveller/IvanHerman">
        dopplr
      </a>
    </li>
    <li>
      <a rel="foaf:holdsAccount"
        href="http://www.facebook.com/ivan.herman">
        Facebook
      </a>
    </li>
  ...
```

using the definition of @about this code yields:

```
<http://www.ivan-herman.net/foaf#me>
  foaf:holdsAccount
    <http://www.dopplr.com/traveller/IvanHerman> .
<http://www.ivan-herman.net/foaf#me>
  foaf:holdsAccount
    <http://www.facebook.com/ivan.herman> .
```

However, RDFa provides the ability to take the common predicate (the @rel value) out into an enclosing element to avoid repeating it. The RDFa code could have been written as:

```
<div about="http://www.ivan-herman.net/foaf#me">
...
  <ul rel="foaf:holdsAccount" >
    <li>
      <a href="http://www.dopplr.com/traveller/IvanHerman">
        dopplr
      </a>
    </li>
```

```
<li>
  <a href="http://www.facebook.com/ivan.herman">
    Facebook
  </a>
</li>
```

...

generating the very same set of triples, but in a more economical (and therefore less error-prone) way. This is, again, not unlike the simplification possibilities offered by Turtle; indeed, the same triples could have been written as:

```
<http://www.ivan-herman.net/foaf#me>
  foaf:holdsAccount
    <http://www.dopplr.com/traveller/IvanHerman>,
    <http://www.facebook.com/ivan.herman> .
```

5.1.3.5 Typing

One of the important patterns in RDF is the usage of RDFS or OWL classes with typing. For example, the FOAF vocabulary includes the notion of a `Person` and, usually, the FOAF data themselves contain a triple of the form:

```
<http://www.ivan-herman.net/foaf#me> rdf:type foaf:Person.
```

This can be expressed in RDFa using the `rdf:type` predicate explicitly, for example,

```
<h1 about="http://www.ivan-herman.net/foaf#me"
  rel="rdf:type" resource=" [ foaf:Person] ">Ivan Herman</h1>
```

would generate the triple above. (Remember that the `[and]` notation is used to denote a CURIE when the attribute value must otherwise be a URI. As `@resource` is close to `@href` and the value of `@href` is defined as URI in XHTML, `@resource` inherits this restriction.)

Because this typing triple is a recurring pattern in RDF and, therefore, in RDFa, a special attribute called `@typeof` is introduced to make this easier. Using this attribute, the XHTML code above could be rewritten as:

```
<h1 about="http://www.ivan-herman.net/foaf#me"
  typeof=" foaf:Person ">Ivan Herman</h1>
```

One of the advantages of using `@typeof` is that it becomes easier to combine typing with other RDFa attributes. For example:

```
<h1 about="http://www.ivan-herman.net/foaf#me"
  typeof=" foaf:Person " property=" foaf:name ">Ivan Herman</h1>
```

succinctly generates two triples, namely:

```
<http://www.ivan-herman.net/foaf#me>
  rdf:type foaf:Person ;
  foaf:name "Ivan Herman" .
```

5.1.3.6 Local (Blank) Nodes

Yet another important concept of RDF is blank nodes. It is not necessary to go into the precise mathematical specification of blank nodes here; suffice it to say that blank nodes represent local nodes in an RDF graph that are not merged with any other nodes in another graph even if their names happen to coincide. (RDF environments are supposed to perform an internal renaming of the nodes in such cases.)

RDFa allows the usage of the `_:x` symbols (where `x` can be any string) to denote a blank node wherever a compact URI (CURIE) is used. This is, again, similar to the Turtle serialization that uses a similar syntax to denote blank nodes. For example, the following RDFa code:

```
<ul about="http://www.ivan-herman.net/foaf#me" rel="foaf:knows">
  <li resource="[_:a]" typeof="foaf:Person">
    <span property="foaf:name">Ben Adida</span>
  </li>
```

(Note again the `[and]` that surround the CURIE in `@resource` to make it distinct from a “real” URL.) The generated triples, in Turtle, may be an almost verbatim translation:

```
<http://www.ivan-herman.net/foaf#me> foaf:knows _:a .
_:a rdf:type foaf:Person;
  foaf:name "Ben Adida" .
```

The RDFa code can include several blank node names, something like:

```
<ul about="http://www.ivan-herman.net/foaf#me" rel="foaf:knows" >
  <li resource="[_:a]" typeof="foaf:Person">
    <span property="foaf:name">Ben Adida</span>
  </li>
  <li resource="[_:b]" typeof="foaf:Person">
    <span property="foaf:name">Mark Birbeck</span>
  </li>
```

yielding two different blank nodes. This type of pattern is fairly frequent, so RDFa includes yet another extra rule regarding the `@typeof` attribute that makes the generation of such graphs easier. Although the exact rules are a little bit complicated, the essence is that by simply removing `@resource` (which is used to generate an internal identifier anyway) one could write:

```
<ul about="http://www.ivan-herman.net/foaf#me" rel="foaf:knows" >
  <li typeof="foaf:Person">
    <span property="foaf:name">Ben Adida</span>
  </li>
  <li typeof="foaf:Person">
    <span property="foaf:name">Mark Birbeck</span>
  </li>
```

This encodes exactly the same RDF data: the `@typeof` attribute without the `@resource` (or `@href`) will automatically generate its own, local, blank node. Note, again, the similarity to a possible, more compact Turtle encoding of these triples:

```
<http://www.ivan-herman.net/foaf#me> foaf:knows
  [a foaf:Person; foaf:name "Ben Adida"],
  [a foaf:Person; foaf:name "Mark Birbeck"] .
```

Although the full RDFa specification includes some more details not covered here (on the more complex behavior of `@typeof`, `@datatype`, generation of so-called XML literals, etc.), this chapter should provide a good enough overview of what can be achieved with RDFa in combination with XHTML. For further details, the interested reader should refer to the RDFa Recommendation [7] or the RDFa Primer [17].

5.1.4 GRDDL and RDFa

This chapter presented GRDDL and RDFa as if they were completely independent technologies. However, this is not exactly the case.

It has already been emphasized that GRDDL is not “bound” to Microformats (nor to XHTML, in fact): although Microformats provide us with probably the most important use case for GRDDL, the specification itself is more general.

It provides a standard way to locate a suitable transformation that could be used to transform the underlying XML (i.e., including XHTML) data into RDF. The specification is silent on what that script does exactly, provided it produces proper RDF. This means that it is perfectly possible to use GRDDL to extract triples from an RDFa+XHTML file: all that is needed is to have a suitable transformation available and to add the necessary markup to the RDFa+XHTML file to use this transformation via GRDDL.

Such XSLT transformation does indeed exist; users wishing to use it may do so by adding the necessary GRDDL markup to their RDFa+XHTML file (see [18] for details). Actually, even this may not be necessary: indeed, the namespace document for XHTML (<http://www.w3.org/1999/xhtml/>) is already prepared for indirect GRDDL processing (and the XHTML namespace is usually present on the `html` element of XHTML files). Truth must be said, however, that at the time of finalizing this manuscript, not many GRDDL implementations implement the indirect approach. That is, using the simple GRDDL markup, as described in [18], might be safer for the time being.

5.2 Example Applications

Microformats have achieved notable use in a number of “Web 2.0” applications, including supporters such as Microsoft and Apple. RDFa, though more complex, has also achieved use in a number of particularly exciting scenarios. Oftentimes, applications choose to support both Microformats and RDFa when the vocabularies are simple enough to allow it. This chapter gives some examples.

5.2.1 Connections Between the Human and Machine-Readable Web

As already referred to in the introduction, one of the main issues for the Semantic Web (or a “Web of Data” in this context) is the availability of data. Beyond the more traditional sources of information (on-the-fly access to databases, XML files, etc.) one of the most important sources is the vibrant world of Web pages, with the dynamic nature provided by such typical Web 2.0 tools as blogs, microblogs, social sites, etc. However, bridging the gap between the “traditional” and the Semantic Web has always been a major source of controversy and difficulties. Indeed, it is difficult, if not impossible, to expect website authors, bloggers, contributors to various Web 2.0 sites, or even automated tools to produce, alongside the (X)HTML pages, separate metadata in a different format like RDF/XML or Turtle. The barriers are not only technical, but also “social”: many website designers, content providers, etc., do not have clear notions of such concepts as metadata, vocabularies, etc. This in spite of the fact that new applications (such as the ones described in more details later in this section) provide very tangible values based on the usage of such additional data.

Both Microformats and RDFa have a major role to play in solving this problem. They offer technologies whereby the same source of information can be used both by the human and the machine; it offers new perspectives to technologies that can generate such additional information either automatically or with a very little help from the users. More importantly, they offer a very easy transition path for existing tools to provide richer content; there is no need for the generation of separate files on a website, separate entries in the database of a CMS system, etc.

A good example of this evolution is a CMS system like Drupal. This open-source tool has helped many organizations to build their online presence and publish content on the Web, such as the White House or the World Bank. With an estimated half million websites running on it, Drupal is ranked among the first three CMS platforms in use today.

Drupal (<http://www.drupal.org>) has recently incorporated Semantic Web structures in general, and RDFa in particular, into its core engine. Drupal 7’s internal data structures are mapped to RDF by default and its theme layer – responsible for rendering the HTML elements – injects RDFa markup in the Web page. Fields exposed by this RDFa are title, date of publication, author information, main content, as well as the possible comments

on the page [19]. This is done automatically for all users and pages without any prior knowledge of RDFa (just as the average user does not know HTML markup). Other CMS services (like Liferay, <http://www.liferay.com/>) are also looking at Semantic Web technologies and, in particular, at RDF to add additional meta information to their Web pages, and it can be expected that some sort of microformat or RDFa will be used by those, too.

Another automatic source of adding microformat and/or RDFa information directly into the pages are plug-ins or additional tools that can be used in conjunction with, say, blogging environments. For example, a plug-in exists for the popular WordPress platform (<http://wordpress.org/extend/plugins/wp-rdfa/>) to automatically add FOAF Dublin Core tags to blog entries using RDFa. The services provided by tools like Zemanta (<http://www.zemanta.com>), Open Calais' Marmoset (<http://www.opencalais.com/documentation/calais-marmoset>), or Nstein (<http://www.nstein.com>) go one step further: these tools analyze the textual content of, for example, a blog entry and propose common tags (based, e.g., on DBpedia terms) by adding microformat or RDFa to the content. These tools can be used either via dedicated APIs or, for example, through plug-ins to blogging environments like WordPress, Movable Type, or even CMS systems like Drupal. (It is interesting to note that there is even a recent initiative to agree on a common tagging format using RDFa, see <http://commontag.org/Home>.)

Tools like Open Calais or Zemanta rely on fairly complex natural language processing techniques. The appearance of Microformats and of RDFa has provided a long-awaited mechanism, whereby the results of such technologies can be finally exploited on the Semantic Web. To put it succinctly, Microformats and RDFa contribute in making the gap between the human and the machine-readable Web eventually disappear.

5.2.2 Search

The Semantic Web has long proposed the idea that, with explicit semantics, search engines may become smarter. With Microformats and RDFa, this evolution is beginning.

5.2.2.1 Yahoo! SearchMonkey and BOSS

In the summer of 2008 Yahoo! released SearchMonkey (<http://developer.yahoo.com/searchmonkey/>) [20, 21], an API that enables developers to customize the presentation of Yahoo!'s search results. Each developer's contribution is packaged as a SearchMonkey application, which users can choose to add to their Yahoo! profile to affect the presentation of their search results. Each SearchMonkey application is given access to the Microformats and RDFa embedded within the individual search result page, which it can use to provide enhanced results.

A number of microformat vocabularies as well as RDFa are also directly searchable via Yahoo!, for example, a search term of the form

```
searchmonkeyid:com.yahoo.page.uf.hcard
```

would return all pages that use the hCard microformat. Similar search terms can be used to detect RDFa in a page. Though the detailed semantic data are not yet indexed and thus are not yet searchable directly, the presence of Microformats or RDFa can be used to filter the results.

In early 2009, Yahoo! enabled SearchMonkey support in BOSS, the Build-Your-Own-Search-Service product where developers can directly call into the Yahoo! search API. Using this service, developers can develop vertical-specific search engines that utilize embedded semantic data to provide rich results.

Of course, the desired next step is the ability to alter the results themselves based on the semantics, not just their presentation. At this point, only prototypes of these search engines exist. One hopes that Yahoo!'s SearchMonkey will evolve to provide this full-fledged semantic search.

5.2.2.2 Google

In May 2009, Google also announced their plans to use Microformats and RDFa information embedded in Web pages [22]. At first, this information is used by Google to enrich the “snippets,” that is, the information the search result listing provides on each entry. Using the Microformat or the RDFa information embedded in the data, the Google search result is able to provide, for example, price ranges and review references to a result on a specific restaurant. In May 2010, this service was extended to what Google calls “Google Squared,” which provides even more information on each search result (images as well as text, the exact origin of specific information, etc). Google has also announced, in September 2009, that they would also index Yahoo!'s SearchMonkey RDFa for video (<http://developer.search.yahoo.com/help/objects/video>, a simple RDFa-based annotation of video embedded in XHTML).

All these are of course the early steps, but it can be expected that embedded semantic data will be used more extensively both by Yahoo! and by Google (and probably by other search engines) in future.

5.2.3 Facebook's Open Graph Protocol

In May 2010 Facebook announced their Open Graph Protocol [23, 24]. The goal of this protocol is to enable any Web page to become a rich object (i.e., a node) in a social graph. A typical usage of this protocol is to add a Facebook “like” button on a page which enables users to make connections to this page and share content back to their friends.

For a page to be part of the graph, it has to contain some metadata. These metadata are added to the page using RDFa [24]. At the time of finalizing this manuscript the RDFa statements appear only in the header of the page and use a few simple terms only, but it is probable that this mechanism will evolve toward more complex vocabularies and usage patterns. A few days after the publication of the protocol a number of sites (e.g., Rotten Tomatoes, NHL, Microsoft, a number of online journals and magazines like the *TIME*) have begun publishing their metadata in RDFa. Some of these sites (e.g., Rotten Tomatoes) have immediately gone beyond the simple Open Graph vocabulary and have added more complex metadata to their pages. In other words, the introduction of this protocol has dramatically increased the information available, if necessary, in RDF thanks to RDFa.

It is interesting to see that, as a result of Yahoo!’s and Google’s usage of embedded semantics and, more recently, due to Facebook’s Open Graph Protocol, a number of sites have begun to automatically include Microformats or RDFa as part of their Web pages. Typical examples for RDFa usage are the individual slide pages on SlideShare (<http://www.slideshare.net/>), the publication of the *London Gazette* (<http://www.london-gazette.co.uk/>), or the product pages of TESCO (<http://www.clothingattesco.com>) or those of BestBuy (<http://www.bestbuy.com>). Google’s map service (<http://maps.google.com>) uses the hCard microformat for contact information, which is also used by Twitter (<http://www.twitter.com>) alongside with some other Microformats.

5.2.4 Browser Extensions

An important application of HTML-embedded metadata such as Microformats and RDFa is the resulting ability to augment the Web-browsing user experience. In particular, semantic data expressed in a Web page can be used by a semantically aware Web browser to connect the current data to past public data seen on other pages or to the user’s private data store. These connections may range from simple “related information” linking to complete archival, private search, and reasoning on the stored information.

5.2.4.1 BlueOrganizer

Adaptive Blue (<http://www.adaptiveblue.com/>) provides a browser add-on. When the add-on encounters a Web page describing products for purchase using Microformats, it connects this content to possible locations where these products can be purchased. For example, a book reviewed on a blog can be automatically linked to its purchase page on Barnes & Noble, Amazon, and others, using the right-click triggered contextual menu. Rather than letting the publisher of the data decide which specific store to link to, the viewer can decide to use their store of choice for the product in question.

5.2.4.2 Operator

Operator (<https://addons.mozilla.org/firefox/4106/>) is a browser add-on that detects Microformats and RDFa and displays a number of data-dependent menu items that allow the user to take quick actions based on the embedded semantic data. A user is able to save an hCard to his/her address book and an hCalendar event to his/her calendar. A user may also look up where to purchase a song marked up using audio RDFa.

Where BlueOrganizer provides a fixed set of actions which they control, Operator is extensible. Developers can write data-dependent actions known as Operator Scripts, and users can choose to install the Operator Scripts they choose.

5.2.4.3 Fuzz

Fuzz (<http://rdfa.digitalbazaar.com/fuzz/trac/>) is a browser add-on that detects and parses RDFa using `librdfa`, a high-speed cross-platform RDFa parser. By default, Fuzz displays Friend-Of-A-Friend (FOAF) [14] information. In general, it can be extended to display any vocabulary, or to take any action based on this vocabulary. Digital Bazaar, the creators of Fuzz, are developing an advanced browser-based music-exchange application built on Fuzz.

5.2.5 Creative Commons and the CC Network

A particularly interesting use case for embedded semantic data in (X)HTML is that of loosely coupled mash-ups – structured data on one web page can be dynamically connected to another application, either on the server side or on the browser side. Applications can connect opportunistically; by publishing a particular vocabulary, one website can automatically be connected to another that seeks to consume that vocabulary. There are no domain-specific APIs – the only connection is the common vocabulary.

Creative Commons has long advocated the use of RDFa to mark up digital works and their many properties, including notably their copyright license [25]. For users who become members of the Creative Commons Network, a detailed XHTML+RDFa fragment is provided for each Web-based work that they wish to claim. This fragment contains a link to the appropriate Creative Commons Deed and a link to their Creative Commons Network profile. Both of these links are typed with RDFa. When a visitor to the work clicks the link to the Creative Commons Deed, the Deed checks its HTTP referrer, parses the RDFa, and displays the appropriate attribution name and URL for the work, as well as a new XHTML+RDFa fragment that can be immediately used, when redistributing or modifying the work, to give credit to the original author.

While the Creative Commons Network plays the role of a digital works registry, other registries can just as easily plug into the Creative Commons Deed, simply by reusing the same RDF vocabulary. By feeding purely off the RDFa, the Creative Commons Deed is a loosely coupled Web application, where any publisher who wishes to connect can do so with ease (► [Fig. 5.3](#)).

Under the following conditions:



Attribution — You must attribute this work to [Ben Adida](#) (with link).

Attribute this work:

```
<div xmlns:cc="http://creativecommons.org/ns#" about="http://ben
```

■ Fig. 5.3

A portion of the Creative Commons Deed, which displays the attribution information because it has parsed the RDFa from its referrer URL and determined to whom credit should be given

5.2.6 Separating User Interface from Raw Content

Another fascinating use case of embedded semantic data is the ability for Web publishers to compartmentalize the development of raw content from advanced visualization. Talis and the University of Plymouth demonstrated just how to do this by developing an advanced JavaScript user interface that feeds off RDFa markup in a Web page [26]. In their own words:

- ▶ The interface to build or edit lists uses a WYSIWYG metaphor implemented in Javascript operating over RDFa markup, allowing the user to drag and drop resources and edit data quickly, without the need to round trip back to the server on completion of each operation. The user's actions of moving, adding, grouping or editing resources directly manipulate the RDFa model within the page. When the user has finished editing, they hit a save button which serialises the RDFa model in the page into an RDF/XML model which is submitted back to the server.

Thus, the markup with its embedded semantics feeds both the visual interface and the back-end data store.

5.2.7 Decentralized Data Distribution Using UK Government Websites

The Central Office of Information is responsible for charting the direction of many of the UK government's websites, and a recent project involved creating a centralized location where the public can search job vacancies that are available in the public sector.

Rather than trying to tackle the enormous task of getting each department to enter their vacancy information into a centralized database, the COI instead took a decentralized approach, allowing each department to retain their existing databases and content management systems; the only change required was that each website should now include RDFa in their pages, in order to describe the jobs available. This RDFa could then be consumed by a central server.

Using RDFa in this way ensures that departments of any size can have their vacancies included in the central store, regardless of whether they have an elaborate CMS, or post

jobs by manually creating HTML. But in addition it creates the possibility that other services can also read this information and use it in a more targeted way (🔗 Fig. 5.4).

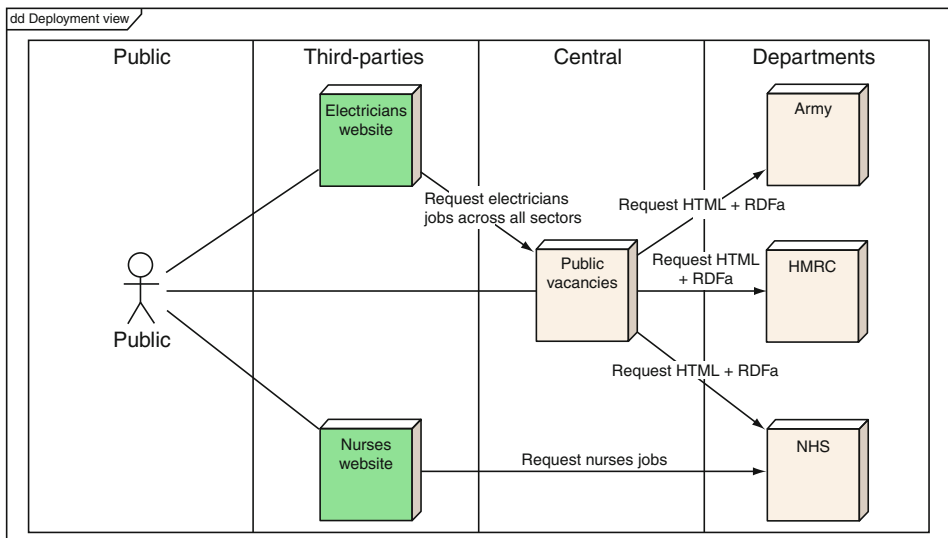
This ability for third-parties to also consume the information is particularly important in another UK government project which uses RDFa, that of consultations.

Consultations are a process where a government department issues one or more documents and then invites interested parties to provide feedback. Since any department could issue a consultation on their website, then it can be difficult for the public to find them, unless they know what they are looking for. Since consultation is an essential part of a modern democracy, a number of sites have sprung up that “screen-scrape” the information, but there is always a problem of accuracy.

With RDFa though, each department can mark up their consultations, and, as with vacancies, make this information available to outside services. And since, as shown in an earlier section, search engines such as Google and Yahoo! are starting to index RDFa, a virtuous circle is formed as information becomes even easier for the public to find and make use of.

5.2.8 Publication of Vocabularies and Datasets

Large vocabularies or RDF Data, in RDF(S), OWL, or SKOS are being published on the Web. One of the problems users face is that, in some cases, the size of the vocabulary as a whole is relatively large, which makes it difficult to consult individual terms online. Although these vocabularies are usually downloadable as one or more files encoded in, say,



🔗 Fig. 5.4

RDFa can be used to publish from departments to central servers, and from there to third-party websites

RDF/XML, what the user often wants is simply to get information on a single individual term without the obligation of storing a large file on his or her local disk.

As an example, the Library of Congress (LoC) of the USA has recently made its Authorities and Vocabularies available in RDF, using SKOS [27] to organize the terms and concept hierarchies (<http://id.loc.gov/authorities/search/>, close to 350,000 concepts are published this way). The importance of this publication is that the Semantic Web community has access to a large number of stable URIs for concepts as different as Semantic Web, or Historical Novel. As an example, the URI:

```
http://id.loc.gov/authorities/sh2002000569#concept
```

can be considered as a reliable reference for the abstract concept for “Semantic Web.” The existence of such URIs, and the fact that the corresponding concepts are properly organized, is extremely important for Semantic Web applications. For example, the concept of “World Wide Web” can be considered as a broader concept than the “Semantic Web”; this fact is duly represented using the SKOS term `skos:broader` binding the two corresponding URIs in the LoC dataset. Applications may then make use of these relationships in organizing and integrating their own data.

To help end users, the LoC has set up its website in such a way that each concept URI is redirected to a XHTML page, giving some of the details on a specific concept (see, e.g., [Fig. 5.5](#)). That is fine for humans. However, to use the RDF/SKOS triples in an application the developer may have to download the whole dataset (about 34 MB of RDF/XML data) and incorporate it in his/her application. This is obviously not ideal; it may lead to versioning issues, for example.

To facilitate this problem, the XHTML pages on the LoC website use RDFa. Indeed, each page on a particular concept contains all the relevant triples encoded in RDFa. This means that an application can, in real time, access those triples through an RDFa-aware tool, instead of maintaining a local copy of the whole dataset; such a tool will then provide triples like:

```
<http://id.loc.gov/authorities/sh2002000569#concept>
  a skos:Concept ;
  dct:created "2004-04-07T00:00:00-04:00"^^xsd:dateTime ;
  skos:broader
    <http://id.loc.gov/authorities/sh95000541#concept>, ... ;
  skos:prefLabel "Semantic Web"@en ;
  ...
```

(where the URI `...sh95000541#concept` refers to another concept, in this case that of the World Wide Web).

The Library of Congress is not the only vocabulary or dataset published this way. A similar example is the STW Thesaurus for Economics (<http://zbw.eu/stw>), published by the German National Library of Economics (considered to be the World’s largest economics library) [28]. Yet another example is DBpedia (<http://www.dbpedia.org>). This dataset is, essentially, a periodic RDF dump of much of the data in Wikipedia enriched by additional links and vocabularies, and made available as Linked Open Data

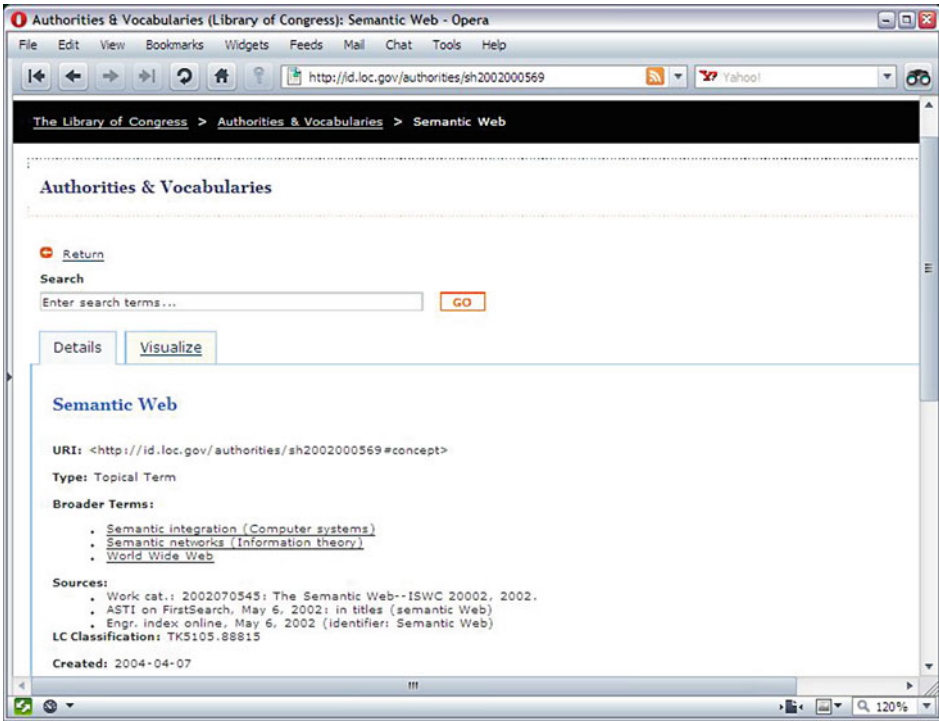


Fig. 5.5
Library of Congress Concept page on Semantic Web

in RDF [29]. While the DBpedia dataset can of course be downloaded, its size makes such a download prohibitive for many applications. However, just like in the LoC or the STW cases, DBpedia URIs can be accessed via a traditional browser, too, with the URIs redirected to XHTML+RDFa pages that contain all the relevant triples for that particular URI. A similar approach is used by dbpedia lite (<http://dbpedia-lite.org>) which, in contrast to DBpedia, is not a dump of the Wikipedia dataset but rather a lightweight layer on top of Wikipedias API providing immediate, though for more modest amount of data in RDF.

5.2.9 Self-documenting Vocabulary Specification

One of the well-known problems in software engineering in general is the proper documentation of code. On the Semantic Web, a similar problem occurs when publishing vocabularies: on the one hand, the vocabulary specification must provide a documentation for humans to read and understand; on the other hand, the same vocabulary must have a machine-readable version available in RDF (using, e.g., RDFS, OWL, or SKOS). The traditional approach is to maintain two files side by side; the synchronization between the human-readable and machine-readable format is left to the author(s) of the vocabulary.

RDFa gives the possibility to merge these two. By using XHTML+RDFa, the author can directly encode the RDF triples of the vocabulary definition into the documentation itself. RDFa-aware tools can then extract the RDF content, while humans can read the documentation in their browser. The Biological Taxonomy Vocabulary (<http://ontology.es/biol/ns>) or Creative Commons ccREL specification [25] (<http://creativecommons.org/ns#>) are good examples.

5.3 Related Resources

The Microformats community maintains its own site (<http://microformats.org>), where the specifications for all individual Microformats as well as the underlying microformat design principles are made available. Individual microformat specifications often have a reference to example usages on the Web (see, e.g., the “hCard Examples in the Wild” (<http://microformats.org/wiki/hcard-examples-in-wild>) page). A number of Microformat tools are also available, both for creation and parsing, and listed on the site (<http://microformats.org/code-tools/>). Finally, a book on Microformats has also been published [30].

Beyond the specification itself [6], W3C’s GRDDL Working Group has also published a GRDDL Primer [13]. The community maintains a Wiki page on GRDDL implementations (<http://esw.w3.org/topic/GrddlImplementations>) and W3C also provides a general GRDDL Service (<http://www.w3.org/2007/08/grddl/>) that can be used to extract RDF from a suitably prepared XML file.

Just as in the case of GRDDL, the relevant W3C Working Group has not only published the formal specification of RDFa [7] but it has also provided an RDFa Primer [17]. The RDFa community maintains its own information site (<http://rdfa.info>), which explains how to use RDFa in a number of specific cases. Tutorials at various conferences are also available online (e.g., [31]). The RDFa info pages include information on various tools for publishing RDFa as well as for parsing and processing.

5.4 Future Issues

The future of the Semantic Web almost certainly relies in large part on its deployment within the existing clickable Web. Microformats and RDFa are two important technologies that will enable this transition to happen. Importantly, both technologies were built to interfere as little as possible with their carrier signal, (X)HTML. As the development effort on HTML5/XHTML5 comes to fruition, it will be possible to include both metadata markup approaches in HTML5 without interfering with HTML’s new features.

Note that at the time of finalizing this manuscript, a new RDFa Working Group is already active in defining a slightly updated version of RDFa (referred to as RDFa 1.1). The new developments do not change the nature of RDFa; the goal is rather to simplify the task of RDFa authors in defining easily, for example, commonly used prefixes. (See a recent blog [32] that describes the most important new features.) The HTML Working Group of

W3C has also published a so-called First Public Working Draft on a document outlining the usage of RDFa in HTML5 and XHTML5. While that draft is based on the current version of RDFa, the plan is to harmonize the future evolution of RDFa 1.1 with the final version of HTML5. Finally, the RDFa Working Group has already published a draft for an RDFa API, that is, a programming language interface whereby Web applications may use the RDFa content embedded in a page directly. (See another recent blog [33] giving a short overview of the API features.)

Over time, one can expect new tools to make use of this embedded data to help Web users connect the data they browse more easily. Tools should appear that easily enable the extraction of significant data from one Web page and connect it to other Web pages via their respective structure. One tool that is beginning to provide these features in a prototype environment is Mozilla's Ubiquity (<http://labs.mozilla.com/projects/ubiquity/>), which enables users to dynamically combine various Web Services based on the content they are seeing, for example, map five selected Craigslist apartments on Google Maps (or on Yahoo! Maps if that is the user's preferred option). One can hope to see these types of applications embrace the interoperable structured data stored within Web pages, using Microformats or RDFa.

Finally, another line of development is to use the RDFa approach more widely to non-HTML-related XML dialects. SVG Tiny 1.2 [8] and Yahoo!'s MediaRSS [9] were already mentioned as precursors of this evolution. Another notable example is the inclusion of RDFa-like attributes into the next version of the OpenDocument format (version 1.2) [34], the file format used, for example, by the OpenOffice.org office application. The upcoming RDFa 1.1 version will make this evolution explicit, and will be defined in such a way that any XML dialect will be able to make use of RDFa attributes.

5.5 Cross-References

- eGovernment
- Semantic Annotation and Retrieval: RDF
- Semantic Annotation and Retrieval: Web of Data

References

1. Beckett, D. (ed.): RDF/XML syntax specification (revised), W3C Recommendation. <http://www.w3.org/TR/rdf-syntax-grammar/> (2004)
2. Beckett, D., Berners-Lee, T.: Turtle – terse RDF triple language, W3C Team Submission. <http://www.w3.org/TeamSubmission/turtle/> (2008)
3. Prud'hommeaux, E., Seaborne, A. (eds.): SPARQL query language for RDF, W3C Recommendation. <http://www.w3.org/TR/rdf-sparql-query/> (2008)
4. Microformats: <http://microformats.org>
5. Clark, J. (ed.): XSL Transformations (XSLT), W3C Recommendation. <http://www.w3.org/TR/xslt> (1999)
6. Connolly, D. (ed.): Gleaning Resource Descriptions from Dialects of Languages (GRDDL), W3C Recommendation. <http://www.w3.org/TR/grddl/> (2007)
7. Adida, B., Birbeck, M., McCarron, S., Pemberton, S. (eds.): RDFa in XHTML: syntax and processing,

- W3C Recommendation. <http://www.w3.org/TR/rdfa-syntax> (2008)
8. Andersson, O., Berjon, R., Dahlström, E., Emmons, A., Ferraiolo, J., Grasso, A., Hardy, V., Hayman, S., Jackson, D., Lilley, C., McCormack, C., Neumann, A., Northway, C., Quint, A., Ramani, N., Schepers, D., Shellshear, A. (eds.): Scalable Vector Graphics (SVG) tiny 1.2 specification, W3C Recommendation. <http://www.w3.org/TR/SVGTiny12/> (2008)
 9. RSS Advisory Board Media RSS specification version 1.1.2. <http://search.yahoo.com/mrss>
 10. hCalendar – microformats wiki. <http://microformats.org/wiki/hcalendar>
 11. Hayes, P. (ed.): RDF semantics, W3C Recommendation. <http://www.w3.org/TR/rdf-mt/> (2004)
 12. W3C: Custom RDF dialects – ESW wiki. <http://esw.w3.org/topic/CustomRdfDialects>
 13. Halpin, H., Davis, I.: GRDDL primer, W3C Working Group Note. <http://www.w3.org/TR/grddl-primer/> (2007)
 14. Brickley, D., Miller, L. (eds.): FOAF vocabulary specification 0.91. <http://xmlns.com/foaf/spec/> (2007)
 15. Biron, P.V., Malhotra, A. (eds.): XML schema part 2: datatype, 2nd edn., W3C Recommendation. <http://www.w3.org/TR/xmlschema-2/> (2004)
 16. DCMI Usage Board: DCMI metadata terms. <http://dublincore.org/documents/dcmi-terms/> (2008)
 17. Adida, B., Birbeck, M. (eds.): RDFa primer, W3C Working Group Note. <http://www.w3.org/TR/xhtml-rdfa-primer/> (2008)
 18. Gandon, F.: Profile for latest GRDDL transformation for RDFa. <http://ns.inria.fr/grddl/rdfa/> (2008)
 19. Corlosquet, S., Delbru, R., Clark, T., Polleres, A., Decker, S.: Produce and consume linked data with Drupal! In: Proceedings of the Eighth International Semantic Web Conference (ISWC 2009), Chantilly. Lecture Notes in Computer Science, vol. 5823, pp. 763–778. Springer, Berlin (2009)
 20. SearchMonkey: <http://developer.yahoo.com/searchmonkey/>
 21. Mika, P.: Improving web search using metadata. <http://www.w3.org/2001/sw/sweo/public/UseCases/yahoo/> (2008)
 22. Goel, K., Guha, R.V., Hansson, O.: Introducing rich snippets. <http://googlewebmastercentral.blogspot.com/2009/05/introducing-rich-snippets.html> (2009)
 23. Open Graph Protocol. <http://developers.facebook.com/docs/opengraph>
 24. Open Graph Protocol. <http://opengraphprotocol.org/>
 25. Abelson, H., Adida, B., Linksvayer, M., Yergler, N. (eds.): ccREL: the creative commons rights expression language, W3C Member Submission. <http://www.w3.org/Submission/ccREL/> (2008)
 26. Clarke, C., Greig, E.: A linked open data resource list management tool for undergraduate students. <http://www.w3.org/2001/sw/sweo/public/UseCases/Talis/> (2009)
 27. Miles, A., Bechhofere, S. (eds.): SKOS simple knowledge organization system reference, W3C Recommendation. <http://www.w3.org/TR/skos-reference> (2009)
 28. Borst, T., Neubert, J.: Publishing STW thesaurus for economics as linked open data. <http://www.w3.org/2001/sw/sweo/public/UseCases/ZBW/> (2009)
 29. Bizer, C., Lehmann, J., Kobilarov, G., Sren, A., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia – a crystallization point for the web of data. *J. Web Semant.* 7(3), 154–165 (2009)
 30. Allsopp, J.: Microformats, Empowering Your Markup for Web 2.0. Springer, New York (2007)
 31. Hausenblas, M., Herman, I., Adida, B.: RDFa – bridging the web of documents and the web of data (tutorial). In: Proceedings of the Seventh International Semantic Web Conference (ISWC 2008), Karlsruhe. Lecture Notes in Computer Science, vol. 5318. Springer, Berlin. <http://www.w3.org/2008/Talks/1026-ISCW-RDFa/RDFa-ISWC08.html> (2008)
 32. Herman, I.: RDFa 1.1 drafts. <http://ivan-herman.name/2010/04/22/rdfa-1-1-drafts/>
 33. Herman, I.: RDFa API draft. <http://ivan-herman.name/2010/06/08/rdfa-api-draft/>
 34. OASIS: OASIS open document format for office applications (OpenDocument) TC. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=office

6 Semantic Annotation and Retrieval: Web of Data

Tom Heath¹ · Christian Bizer²

¹Talis Systems Ltd, Birmingham, UK

²Freie Universität Berlin, Berlin, Germany

6.1	<i>Scientific Overview</i>	193
6.1.1	Introduction	193
6.1.2	From a Web of Documents to a Web of Data	194
6.1.2.1	Semantic Web	194
6.1.2.2	Web APIs	195
6.1.2.3	Microformats	196
6.1.2.4	Dataspaces	197
6.1.2.5	Linked Data	197
6.1.3	Topography of the Web of Data	198
6.1.4	Media	200
6.1.5	Publications	200
6.1.6	Life Sciences	201
6.1.7	Geographic Data	201
6.1.8	User-Generated Content	201
6.1.9	Cross-Domain Data Sources	201
6.1.10	eGovernment Data Sources	202
6.2	<i>Technical Overview: The Web of Data in Practice</i>	203
6.2.1	URIs, HTTP, and RDF	203
6.2.2	Choosing URIs and RDF Vocabularies	206
6.2.3	Publishing Tools	207
6.2.4	Link Generation	208
6.2.5	Meta-Information About Published Data	209
6.2.5.1	Provenance Meta-Information	210
6.2.5.2	Technical Meta-Information	210
6.2.5.3	Licensing	211
6.3	<i>Example Applications: Retrieving and Consuming from the Web of Data</i>	211
6.3.1	Application Architectures for the Web of Data	211
6.3.2	Web of Data Browsers	213
6.3.3	Web of Data Search Engines and Indexes	214
6.3.3.1	Human-Oriented Search Engines	214

6.3.3.2	Application-Oriented Indexes	216
6.3.4	Web of Data Mash-ups	217
6.3.4.1	Revyu	217
6.3.4.2	DBpedia Mobile	218
6.3.4.3	Talis Aspire	219
6.3.4.4	BBC Programs and Music	219
6.3.4.5	DERI Pipes	219
6.4	<i>Related Resources</i>	220
6.4.1	Key Papers	220
6.4.2	Community Websites	221
6.4.3	Linked Data Events	221
6.5	<i>Future Issues</i>	222
6.5.1	User Interfaces and Interaction Paradigms	222
6.5.2	Schema Mapping and Data Fusion	223
6.5.3	Trust, Quality, and Relevance	224
6.5.4	Link Maintenance	225
6.5.5	Privacy	225

Abstract: The World Wide Web has radically altered the way knowledge is shared by lowering the barrier to publishing and accessing documents. Recent initiatives have extended the principles and architecture of the Web to sharing and accessing data, resulting in an interconnected, global data space – the Web of Data. This chapter reviews the historical and scientific context to the emergence of this Web of Data, before detailing the technologies and principles on which it is based. Applications that have been developed and deployed on the Web of Data are reviewed, as are key resources in the field. The chapter concludes with a discussion of current research challenges in areas such as user interfaces, data fusion, link maintenance, trust, and privacy.

6.1 Scientific Overview

6.1.1 Introduction

The World Wide Web is built upon the idea to set hyperlinks between Web documents. Hypertext links allow users to traverse this global information space using Web browsers, while search engines index the documents and analyze the structure of links between them to infer potential relevance to users' search queries [1]. This functionality has been enabled by the generic, open, and extensible nature of the Web [2], which is also seen as a key feature in the Web's unconstrained growth.

Despite the inarguable benefits the Web provides, until recently the same principles that enabled the Web of documents to flourish have not been applied to data. Traditionally, data published on the Web has been made available as raw dumps in formats such as CSV or XML, or marked up as HTML tables, sacrificing much of its structure and semantics. In the conventional hypertext Web, the nature of the relationship between two linked documents is implicit, as the data format, that is, HTML, is not sufficiently expressive to enable individual entities described in a particular document to be connected by typed links to related entities.

However, in recent years, the Web has evolved from a global information space of linked documents to one where both documents and data are linked. Underpinning this evolution is a set of best practices for publishing and connecting structured data on the Web, known as linked data. Linked data relies primarily on three technologies to connect structured data that are related but stored in different systems or locations: uniform resource identifiers (URIs) [3], the HyperText Transfer Protocol (HTTP) [4], and the Resource Description Framework (RDF) [5]. It is important to note that two of these technologies, URIs and HTTP, have been central to the Web since its inception, meaning that the Web of Data is not a separate, distinct entity, but an additional layer interwoven with the Web of hypertext documents. In addition, the basic value proposition of linked data is the same as for the Web at large – that the value and utility of a resource can be measured by the number and value of incoming links [2]. Therefore, by exploiting the infrastructure of the Web to enable links between distributed data sources, a network effect can be created that encourages further participation.

The result of the adoption of the linked data principles is a global data space that is architecturally accessible to all and to which anyone can contribute. This data space contains and connects data from diverse domains such as people, companies, books, scientific publications, films, music, television and radio programs, genes, proteins, drugs and clinical trials, online communities, statistical and scientific data, and reviews. A number of these domains are examined in more detail below.

The Web of Data enables new types of applications. There are generic linked data browsers that allow users to start browsing in one data source and then navigate along links into related data sources. There are linked data search engines that crawl the Web of Data by following links between data sources and provide expressive query capabilities over aggregated data, similar to how a local database is queried today. The Web of Data also opens up new possibilities for domain-specific applications. Unlike Web 2.0 mash-ups which work against a fixed set of data sources, linked data applications operate on top of an unbound, global data space. This provides the potential to deliver more complete answers as new data sources appear on the Web.

This chapter gives an introduction to the Web of Data, that builds upon and extends existing seminal work in the field [6, 7]. The remainder of this section will give a scientific overview of the topic, with particular emphasis on the historical context in which the Web of Data has emerged. ▶ Section 6.2 will give a technical overview of the standards and best practices that underpin the Web of Data, while ▶ Sect. 6.3 will review applications that build upon the Web of Data. Key resources will be summarized in ▶ Sect. 6.4, before discussion of ongoing and future research challenges.

6.1.2 From a Web of Documents to a Web of Data

The desire to extend the capabilities of the Web to the publishing of structured data is not new, and can be traced back to the earliest proposal for the World Wide Web (<http://www.w3.org/History/1989/proposal.html>) and subsequent papers on the topic [8, 9]. Trends foreseen at these early stages of the Web's existence included “Evolution of objects from being principally human-readable documents to contain more machine-oriented semantic information” [8], which can be seen as the seeds of an idea that became known as the “Semantic Web” [9].

6.1.2.1 Semantic Web

The vision of a Semantic Web has been interpreted in many different ways, for example, Berners-Lee et al. [10] and Marshall and Shipman [11]. However, despite this diversity in interpretation, the original goal of building a Web of machine-readable data remains constant across the original literature on the subject. According to Berners-Lee [9], “The first step is putting data on the Web in a form that machines can naturally understand, or converting it to that form. This creates what I call a Semantic Web – a Web of data that can be processed directly or indirectly by machines.” In the time taken to implement the

Semantic Web vision, a number of other related developments occurred that pertain to the availability of data on the Web, while not creating a Web of Data itself. An overview of these developments is given in the following.

6.1.2.2 Web APIs

In recent years, major Web data sources such as Amazon, eBay, Flickr, and Twitter have provided access to their underlying databases via Web APIs. In the case of early Web companies such as Amazon, the provision of access to the product catalog via an API enabled the emergence of a new ecosystem of smaller traders supported by the Amazon technical and fulfillment infrastructure. Recent additions to the Web landscape, such as Twitter, have provided access to content from an early stage of their existence, via Web APIs. The goal of providing such APIs is to enable ecosystems of participation based on usage of the underlying data in a range of applications and locations, rather than simply relying on Web users visiting a particular destination site. The wealth of data accessible via Web APIs has led to the development of mash-ups that combine data from multiple different sources. The Website ProgrammableWeb.com currently lists over 2,100 such APIs as well as roughly 5,000 mash-ups based on these (September 2010).

Web APIs generally follow one of a range of different paradigms, of which the most widely adopted is probably Representational State Transfer (REST) (<http://sweet-dev.open.ac.uk/war/Papers/mmaWebAPISurvey.pdf>) [12]. In common with the linked data approach, URIs as global identifiers and the HTTP protocol for resource access are central to the REST paradigm.

In response to queries, Web APIs typically return documents that encode data in formats such as XML or JSON. These documents have URIs and are accessed via HTTP; however, this scenario bears some important differences to the linked data paradigm: The methods supported by a particular Web API are generally specific to that API, meaning that a developer wishing to use data via this API must learn both the proprietary methods supported and the structure/schema of the data that are returned in response to that method call.

In addition, while the documents returned by a Web API request have HTTP URIs by which they are uniquely and globally identified, it is rare that the entities referenced in the returned data have the same. For example, an XML document returned in response to a Web API request may describe a book, including attributes such as the author, publisher, and ISBN. While the ISBN may be considered a globally unique identifier for the book (ignoring the few cases where this does not hold), comparable identifiers for the author or the publisher are often not provided. This prevents applications that consume data from multiple sources from easily determining that two documents refer to the same entity. Instead, application-specific logic must be applied to infer co-reference.

Furthermore, even where identifiers are provided for each entity featured in a document, the scope of these identifiers is generally limited to the dataset or provider in question, or at best is domain-specific, as in the case of ISBNs. This introduces a further

limitation, whereby a specialized registry (which in turn likely implements a proprietary API) is required in order to look up that identifier. In summary, such arrangements provide no global identification scheme for entities referenced in snippets of data, no reliable means to make links between related entities described in different datasets or documents, and no source-neutral mechanism for looking up these entities.

The consequence of this situation is that most Web APIs represent isolated silos of data that are accessible over the Web, but that cannot generally be woven into the Web through links to and from related data. As a result, creators of mash-ups based on such APIs typically have to invest considerable effort in application-level entity consolidation (i.e., determining that two documents do in fact refer to the same author), the results of which are presented to a human user in visual form but are rarely propagated back into the Web.

The effort required to integrate in this fashion data from varied Web APIs means that traditional mash-ups based on Web APIs are always implemented against a fixed set of data sources that must be selected a priori. Mash-ups cannot realistically be implemented against all the data that are available on the Web or will become available during the lifetime of the application, due to the manual integration effort. What is required is a common underlying framework for connecting and merging related data available on the Web. More details on semantics applied to Web APIs can be found in [▶ Semantic Web Services](#).

6.1.2.3 Microformats

One response to the desire for greater availability of structured data on the Web has been the Microformats movement (<http://microformats.org/>). The term Microformats refers to a set of simple schemas for embedding certain types of data in HTML documents via the attributes of HTML tags. While the Microformats movement has gained significant momentum, the approach has a number of significant limitations.

Firstly, those Microformats currently available that have reached “Specification” status cover a very limited number of domains, while the remainder have simply “Draft” status. One possible explanation for this is that creating new Microformats requires adherence to a rigid process (<http://microformats.org/wiki/process>) that is not fully open. Creating a new Microformat is not as simple as defining a schema that describes a particular area of interest and publishing this for use by others. Instead the process must be conducted through the Microformats community, which could act as a bottleneck to the development of Microformats for coverage of new domains. One rationale for maintaining a rigid process is to ensure that new properties are not introduced with the same name as those in existing Microformats, as all Microformats share the same namespace.

Secondly, and in common with Web APIs as described above, it is not commonplace for entities described in Microformat-enhanced HTML documents to be given unique identifiers. This prevents assertions about relationships between entities, such as linking an hCalendar event to an hCard data in another document about people involved in the event. In fact, while Microformats often include properties such as “URI” (in hCard, for

example), there is no abstract data model underlying Microformats that distinguishes explicitly between string literals and those that should be treated as URIs identifying related resources and can therefore be dereferenced, as is the case with RDF. More details on Microformats can be found in [▶ Semantic Annotation and Retrieval: Web of Hypertext – RDFa and Microformats](#).

6.1.2.4 Dataspaces

A recent development within the databases community that is very relevant to the Web of Data is the concept of *dataspaces* [13]. Dataspaces provide a target system architecture around which ongoing research on reference reconciliation, schema matching and mapping, data lineage, data quality, and information extraction is unified [13]. In contrast with other information-integration systems, dataspace systems offer best-effort answers before complete semantic mappings are provided to the system. A key idea of dataspaces is that the semantic cohesion of a dataspace is increased over time by different parties providing mappings; the same pay-as-you-go data integration approach that currently emerges on the Web of linked data. The Web of Data can therefore be seen as a realization of the dataspace concept on global scale, relying on a specific set of Web standards in order to be closely aligned with the overall architecture of the Web. It is therefore likely that the Web of Data will benefit a lot from the ongoing research in the databases community on dataspaces.

6.1.2.5 Linked Data

While notable progress had been made in defining and implementing the Semantic Web technology stack in the time since the original vision had been outlined, for many years there was little tangible evidence of the existence of the Semantic Web. A significant catalyst in addressing this situation was the publication of the linked data principles by Tim Berners-Lee in 2006 [14], a set of best practices for publishing and connecting structured data on the Web. In summary, these principles provide guidelines on how to use standardized Web technologies to set data-level links between entities described in different data sources. Therefore, while the Semantic Web, or Web of Data, remains the goal, linked data make up the constituent parts of that Web.

Over time, with linked data as a foundation, some of the more sophisticated proposals associated with the Semantic Web vision, such as intelligent agents, may become a reality. However, in the first instance, linked data provide a more uniform set of mechanisms for data access and integration over the Web. By publishing linked data, data owners can lower the barrier to integration, application, and reuse of data from multiple, distributed and heterogeneous sources. The technical details of linked data are described in detail below. For now it is sufficient to highlight some of the unique aspects of linked data, relative to the alternative approaches discussed above.


By using a small, consistent set of technologies and modes of access, by assigning HTTP URIs to entities described in the data, and by linking these entities using RDF, linked data provide access to a global, unbounded dataspace. This improves the discoverability of relevant data, as data sources can be more easily crawled by search engines and accessed using generic data browsers.


In contrast to mash-ups that utilize a fixed set of data sources, linked data applications can discover new data sources by following RDF links and take advantage of new data sources as they appear on the Web, without needing to change the application code. Therefore, linked data technologies can contribute to connecting the different data silos that currently exist on the Web back into a single global information space. Related data about the same entity from different sources can be aggregated, fused, and queried, in much the same way as a local database is queried today, except over distributed, heterogeneous data.

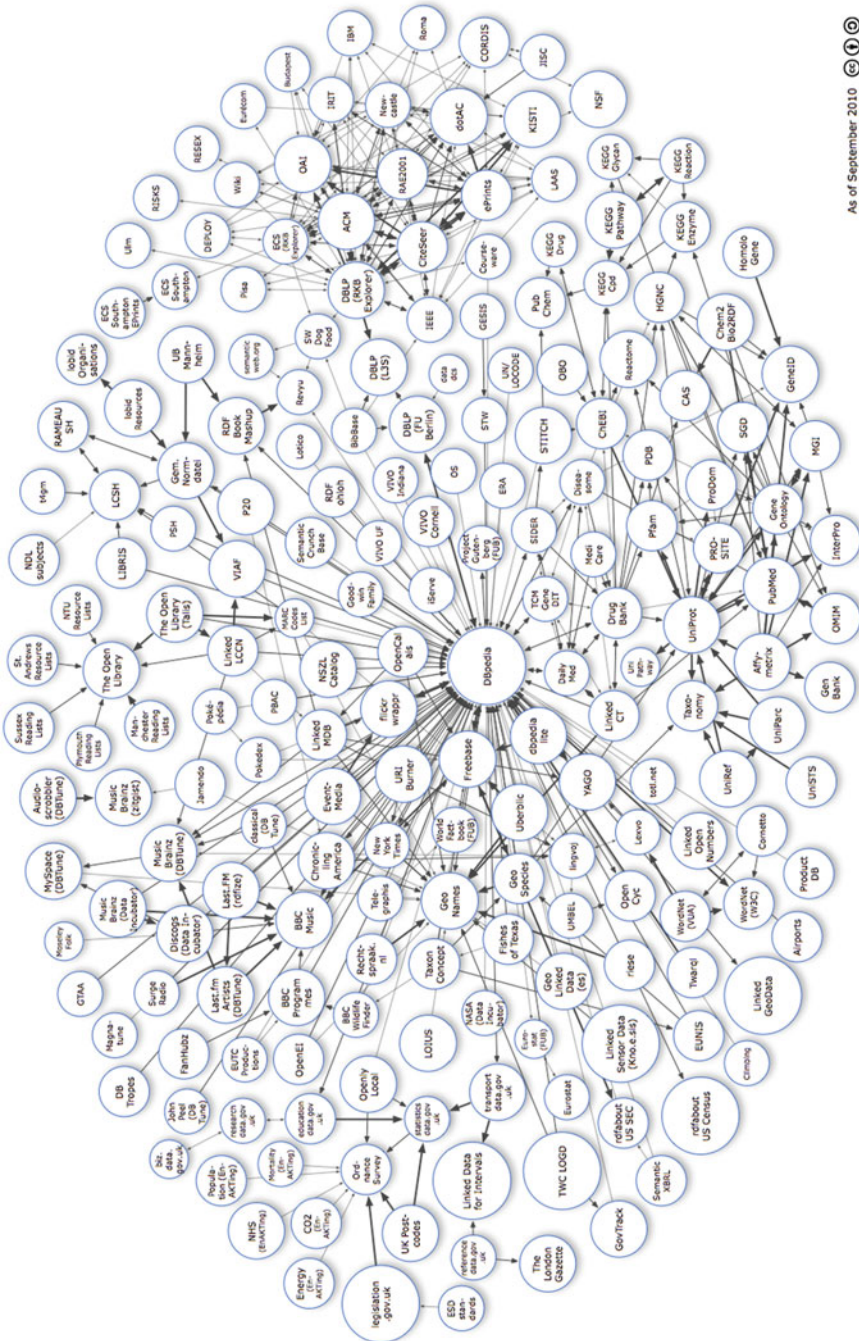
Applications wishing to consume linked data must simply understand the vocabularies or schemas used to describe data (or be able to access mappings from novel to known vocabularies), rather than a wide range of API-specific methods and data formats. In contrast to the Microformats approach, linked data are not limited in the vocabularies that can be used to describe data, and the vocabulary development process itself is completely open; anyone can develop and publish an RDF vocabulary in their own namespace, for use by others.

6.1.3 Topography of the Web of Data

The most visible example of adoption and application of the linked data principles has been the Linking Open Data (LOD) community project (<http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>), a grassroots community effort founded in February 2007 and supported by the W3C Semantic Web Education and Outreach Group. The original and ongoing aim of the project is to bootstrap the Web of Data by identifying existing datasets that are available under open licenses, converting these to RDF according to the linked data principles, and publishing them on the Web.

Participants in the early stages of the project were primarily researchers and developers in university research labs and small companies. Since that time, the project has grown considerably, to include significant involvement from large organizations such as the BBC, New York Times, Library of Congress, and the UK Government. This growth is enabled by the open nature of the project, where anyone can participate simply by publishing a dataset according to the linked data principles and interlinking it with existing linked datasets. An indication of the range and scale of the Web of Data originating from the Linking Open Data project is provided in  *Fig. 6.1*. Each node in this “cloud” diagram represents a distinct dataset published as linked data, as of September 2010.

The arcs in  *Fig. 6.1* indicate that links exist between entities in the two connected datasets. Heavier arcs roughly correspond to a greater number of links between the two linked datasets, while bidirectional arcs indicate that outward links to the other exist in each dataset.



As of September 2010




Fig. 6.1

Linking open data cloud diagram giving an overview of published datasets and their interlinkage relationships

Calculating the exact size of the Web of Data is challenging, due to the fact that much of the data in this cloud are being generated by wrappers around existing relational databases or APIs which first must be crawled before they can be counted or analyzed [15]. Alternatively, the size of the Web of Data can be estimated based on the dataset statistics that are collected by the LOD community in the project wiki. According to these statistics, the Web of Data currently consists of 25 billion RDF triples, which are interlinked by around 395 million RDF links (September 2010) (<http://esw.w3.org/topic/TaskForces/CommunityProjects/LinkingOpenData>).

The Web of Data is diverse in nature, comprising data about geographic locations, people, companies, books [16], scientific publications [17], films [18], music, television and radio programs [19], genes, proteins, drugs and clinical trials [20, 21], online communities, census results, and reviews [22].

In  [Fig. 6.1](#) we can see that a broad range of domains are covered in the datasets in the cloud. The following sections give an overview of the main datasets as well as ongoing data publication efforts. Further details and references to the mentioned datasets are found in the Linking Open Data wiki.

6.1.4 Media

A major linked data publisher within the media industry is the British Broadcasting Corporation (BBC). The BBC/`programmes` and `music` sites provide data about episodes of radio and TV programs, and musical artists, respectively [19]. The data are interlinked with Musicbrainz, an open-license music database, and with DBpedia, a linked data version of Wikipedia. The links between BBC/`music`, Musicbrainz, and DBpedia allow applications to retrieve and combine data about artists from all three sources. The New York Times has recently announced the publication of its subject headings as linked data (<http://data.nytimes.com/>), while intentions to publish linked data have also been indicated by CNET and Thomson Reuters. Thomson Reuters has also developed OpenCalais, a service for the annotation of text document, with URIs from the linked data cloud referring to places, companies, and people.

6.1.5 Publications

The American Library of Congress and the German National Library of Economics publish their subject heading taxonomies as linked data. Linked data about scholarly publications are provided by the L3S Research Center which hosts a linked data version of the DBLP bibliography. The ReSIST project publishes and interlinks bibliographic databases such as the IEEE digital library, CiteSeer, and various institutional repositories. Linked data about books are provided by the RDF Book Mashup, a wrapper around the Amazon and the Google Base APIs. The Open Archives Initiative has based its new Object Exchange and Reuse (OAI-ORE) standard on the linked data principles and it is likely that the deployment of this standard will further accelerate the availability of linked data related to publications.

6.1.6 Life Sciences

A major provider of linked data related to life sciences is the Bio2RDF project which has interlinked more than 30 widely used life sciences datasets including UniProt, KEGG, CAS, PubMed, and the Gene Ontology [20]. Altogether, the Bio2RDF datasets comprise more than 2 billion RDF triples. Within the W3C Linking Open Drug Data effort, the pharmaceutical companies Eli Lilly, AstraZeneca, and Johnson & Johnson cooperate to interlink open-license data about drugs and clinical trials in order to ease drug discovery [21]. For further information about developments in this area we refer the reader to [eScience](#), this volume.

6.1.7 Geographic Data

Geonames, an open-license geographical database, publishes linked data about 8 million locations. The LinkedGeoData project [23] publishes a linked data version of OpenStreetMap providing information about more than 350 million spatial features. Locations in Geonames and LinkedGeoData are interlinked where possible with corresponding locations in DBpedia. The Ordnance Survey, Great Britain's national mapping agency, has started to publish topological information prescribing the administrative areas within the UK as linked data (<http://data.ordnancesurvey.co.uk/>). There are also conversions of the EuroStat, World Factbook, and US Census datasets available as linked data.

6.1.8 User-Generated Content

An increasing amount of metadata about user-generated content from Web 2.0 sites are becoming available as linked data. Examples include the flickr wrapper around the Flickr photo-sharing service and the SIOC exporters for WordPress, the Drupal content management system, and the phpBB bulletin boards. Zemanta provides tools for the semiautomated enrichment of blog posts with data-level links pointing to DBpedia, Freebase, MusicBrainz, and Semantic CrunchBase. A further service for the annotation of Web content with linked data URIs is Faviki. These annotations connect the classic document Web with the Web of Data. The links can be used by applications to retrieve background information about the topics of a blog post or a location depicted by a photo, which can in turn be used by the application to provide a richer user experience. The reviewing and rating site Revyu.com [22], described in more detail below, is a native linked data site with links from reviewed items to various datasets including DBpedia and the OpenGuides [24].

6.1.9 Cross-Domain Data Sources

Data sources that provide information spanning multiple domains are crucial for connecting data into a single global data space and to avoid the fragmentation of the

dataspace into distinct topical islands. An example of such a data source is DBpedia [25], which publishes data that have been extracted from the “infoboxes” commonly seen on the right-hand side of Wikipedia articles. As DBpedia covers a wide range of topics, and has a high degree of conceptual overlap with various other datasets, various data publishers have started to set links from their data sources to DBpedia, making DBpedia one of the central interlinking hubs within the Linking Open Data cloud (cf. [Fig. 6.1](#)). A second major source of cross-domain data is Freebase, developed by Metaweb, a startup recently acquired by Google. Freebase is an open-license database that users can edit in a similar fashion as they edit Wikipedia today. Cross-domain ontologies that are available as linked data include WordNet, OpenCyc, YAGO, and UMBEL. These ontologies are interlinked with DBpedia, which allows applications to mash-up data from all sources.

6.1.10 eGovernment Data Sources

Public organizations produce a wealth of highly relevant data ranging from economic statistics, registers of companies, registers of land ownership, data about local schools, crime statistics, to the voting record of local political representatives. Giving the public easy access to these data enables greater accountability, helps people to make informed choices, and allows third parties to create tools to analyze and work with the data. Many public-sector organizations are required by their mandate to make data resulting from their operations accessible to the general public. In practice however, various barriers, such as proprietary data formats and retrieval mechanisms, hinder access to these data.

In the USA, the Obama administration has begun efforts to address these issues. The Data.gov website was recently launched and currently provides access to 47 datasets generated by the Executive Branch of the Federal Government. These datasets are informally published as linked data by the Data-gov Wiki project (<http://data-gov.tw.rpi.edu/>).

The potential of linked data for overcoming these barriers is increasingly understood. The former British Prime Minister Gordon Brown appointed Tim Berners-Lee as an expert adviser on public information delivery. Berners-Lee has published a Web design note about putting government data online [26] and has worked together with the UK Power of Information Taskforce on realizing these ideas. A first result of this effort is that the UK Civil Service has started to publish open position postings on their websites as linked data [27]. Furthermore, a significantly larger effort is underway to identify datasets within government departments that may be of interest to the public, and release these through the data.gov.uk site (<http://data.gov.uk/>). Of these, a number have already been converted to RDF and published according to the linked data principles, including datasets related to agriculture, schools, traffic flows, and crime surveys.

In order to work more closely with governments and to support public institutions in using open Web standards, the World Wide Web Consortium has formed an eGovernment Interest Group. A first result of the work of this group is the W3C note “Improving Access to Government through Better Use of the Web” [28] which highlights the benefits of open, standard-based access to government data and discusses technical

options to provide such access. For further information about developments in this area we refer the reader to [eGovernment](#), this volume.

6.2 Technical Overview: The Web of Data in Practice

In summary, linked data is simply about using the Web to create typed links between data from different sources. These may be as diverse as databases maintained by two organizations in different geographical locations, or simply heterogeneous systems within one organization that, historically, have not easily interoperated at the data level. Technically, linked data refer to data published on the Web in such a way that it is machine-readable, its meaning is explicitly defined, it is linked to other external datasets, and can in turn be linked to from external datasets.

While the primary units of the hypertext Web are HTML documents connected by untyped hyperlinks, linked data rely on documents containing data in RDF format [5]. However, rather than simply connecting these documents, linked data use RDF to make typed statements that link arbitrary things in the world. The result, which is referred to as the Web of Data, may more accurately be described as a Web of things in the world, described by data on the Web.

Berners-Lee outlined in [14] a set of best practices for publishing data on the Web in a way that all published data become part of a single global data space:

1. Use URIs as names for things.
2. Use HTTP URIs so that people can look up those names.
3. When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL).
4. Include links to other URIs, so that they can discover more things.

These have become known as the linked data principles, and provide a basic recipe for publishing and connecting data using the infrastructure of the Web while adhering to its architecture and standards.

6.2.1 URIs, HTTP, and RDF

Linked data rely on two technologies that are fundamental to the Web: uniform resource identifiers (URIs) [3] and the HyperText Transfer Protocol (HTTP) [4]. While Uniform Resource Locators (URLs) have become familiar as addresses for documents and other entities that can be located on the Web, uniform resource identifiers provide a more generic means to identify any entity that exists in the world.

Where entities are identified by URIs that use the `http://scheme`, these entities can be looked up simply by dereferencing the URI over the HTTP protocol. In this way, the HTTP protocol provides a simple yet universal mechanism for retrieving resources that can be serialized as a stream of bytes (such as a photograph of a dog), or retrieving descriptions of entities that cannot themselves be sent across the network in this way (such as the dog itself).

URIs and HTTP are supplemented by a technology that is critical to the Web of Data: the Resource Description Framework (RDF) [5]. Whilst HTML provides a means to structure and link documents on the Web, RDF provides a generic, graph-based data model with which to structure and link data that describe things in the world.

The RDF model encodes data in the form of *subject, predicate, object* triples. The subject and object of a triple are both URIs that each identify a resource, or a URI, and a string literal, respectively. The predicate specifies how the subject and object are related, and is also represented by a URI.

For example, an RDF triple can state that two people, A and B, each identified by a URI, are related by the fact that A knows B. Similarly an RDF triple may relate a person C to a scientific article D in a bibliographic database by stating that C is the author of D. Two resources linked in this fashion can be drawn from different datasets on the Web, allowing data in one data source to be linked to that in another, thereby creating a Web of Data. Consequently, it is possible to think of RDF triples that link items in different datasets as analogous to the hypertext links that tie together the Web of documents.

RDF links [29] take the form of RDF triples, where the subject of the triple is a URI reference in the namespace of one dataset, while the object of the triple is a URI reference in the other.

➤ *Figure 6.2* contains three example RDF links. The first link states that a resource identified by the URI <http://www.w3.org/People/Berners-Lee/card#i> knows another resource called <http://www.ivan-herman.net/foaf.rdf#me>. When the first URI is dereferenced over the HTTP protocol asking for content type `application/rdf+xml`, the W3C Web server answers with an RDF description of the identified resource, in this case a person called Tim Berners-Lee. When the second URI is dereferenced, the `ivan-herman.net` server provides an RDF graph describing Ivan Herman. Dereferencing the predicate URI <http://xmlns.com/foaf/0.1/knows> yields an RDFS definition of the link type `knows`. The second RDF link states that Tim Berners-Lee is based near a location described in the DBpedia dataset. Dereferencing this URIs yields information about the location, in this case the Cambridge in Massachusetts, USA. The third RDF link connects the description of Tim provided by the W3C server

```
Subject: http://www.w3.org/People/Berners-Lee/card#i
Predicate: http://xmlns.com/foaf/0.1/knows
Object: http://www.ivan-herman.net/foaf.rdf#me
```

```
Subject: http://www.w3.org/People/Berners-Lee/card#i
Predicate: http://xmlns.com/foaf/0.1/based_near
Object: http://dbpedia.org/resource/Cambridge%2C_Massachusetts
```

```
Subject: http://www.w3.org/People/Berners-Lee/card#i
Predicate: http://www.w3.org/2002/07/owl#sameAs
Object: http://www4.wiwiss.fu-berlin.de/dblp/resource/person/100007
```

■ **Fig. 6.2**

Examples of RDF links

with a description of Tim provided by a DBLP bibliography mirror server by stating that the URI <http://www.w3.org/People/Berners-Lee/card#i> and the URI <http://www4.wiwiss.fu-berlin.de/dblp/resource/person/100007> both refer to the same real-world entity.

Retrieved resource descriptions may contain additional RDF links. For instance, dereferencing the URI http://dbpedia.org/resource/Cambridge%2C_Massachusetts yields amongst other information an *owl:sameAs* link stating that Freebase uses the URI <http://rdf.freebase.com/ns/guid.9202a8c04000641f8000000005bbe83b> to refer to the city. By also dereferencing this URI, a client application can retrieve information about the city from a third data source.

The RDF Vocabulary Definition Language (RDFS) [30] and the Web Ontology Language (OWL) [31], provide a basis for creating schemas or ontologies that can be used to describe entities in the world and how they are related. For example, RDFS or OWL could be used to create a sports vocabulary that defines classes of entities such as *Player* and *Team*, and relationships between them such as *playsFor*. These vocabularies are themselves expressed in RDF, using terms from RDFS and OWL, which provide varying degrees of expressivity in modeling domains of interest. Anyone is free to publish schemas to the Web of Data, which in turn can be connected by RDF triples that link classes and properties in one schema to those in another, thereby defining mappings between related schemas.

By employing HTTP URIs to identify resources, the HTTP protocol as retrieval mechanism and the RDF data model to represent resource descriptions, linked data directly build on the general architecture of the Web [2]. The Web of Data can therefore be seen as an additional layer that is tightly interwoven with the classic document Web and has many of the same properties:

- The Web of Data is generic and can contain any type of data.
- Anyone can publish data to the Web of Data.
- Data publishers are not constrained in choice of vocabularies with which to represent data.
- Entities are connected by RDF links, creating a global data graph that spans data sources and enables the discovery of new data sources.

By publishing data on the Web according to the linked data principles, data providers add their data to a global data space, which allows data to be discovered and used by various applications and within various contexts. Publishing a dataset as linked data on the Web involves the following three basic steps:

1. Assign URIs to the entities described by the dataset and provide for dereferencing these URIs over the HTTP protocol into RDF representations.
2. Set RDF links to other data sources on the Web, so that clients can navigate the Web of Data as a whole by following RDF links.
3. Provide meta-information about published data, so that clients can assess the quality of published data as well as its licensing terms.

Examples of RDF links across data sources can be found in examples 2 and 3 in [Fig. 6.2](#), while the reader is referred to the relevant section below for further discussion of providing meta-information about published data.

In the following section, an overview is provided about each of these tasks as well as about tools that have been developed to support publishers with each task.

6.2.2 Choosing URIs and RDF Vocabularies

Data provider can choose between two HTTP URI usage patterns to identify entities: 303 URIs and hash URIs [32]. Both patterns ensure that clients can distinguish between URIs that identify real-world entities and URIs that identify Web documents describing these real-world entities. The chosen pattern determines how URIs are dereferenced by client applications. For 303 URIs, an HTTP 303 redirect is used to redirect the client from a URI identifying a real-world entity to a Web document describing the entity. Retrieving data about a real-world entity therefore involves two separate HTTP calls: one for dereferencing the URI identifying the entity into the URI of a document describing it and a second call for retrieving this document. The hash URI pattern provides an alternative, which allows data to be retrieved in a single HTTP call. A hash URI contains a *fragment*, a special part that is separated from the rest of the URI by a hash symbol (“#”). When a client retrieves a hash URI, then the HTTP protocol requires the fragment part to be stripped off before requesting the URI from the server. Using hash URIs to identify real-world entities therefore allows to retrieve data about the entity directly with one HTTP call without creating ambiguity between the entity and the document [32].

In an open environment like the Web, different information providers publish data about the same real-world entity, for instance, a geographic location or a celebrity. As they may not know about each other, they introduce different URIs to identify the same entity. For instance, DBpedia uses the URI <http://dbpedia.org/resource/Berlin> to identify Berlin, while Geonames uses the URI <http://sws.geonames.org/2950159/> to identify Berlin. As both URIs refer to the same real-world entity, they are called URI aliases. URI aliases are common on the Web of Data, as it cannot realistically be expected that all information providers agree on the same URIs to identify an entity. URI aliases also provide an important social function to the Web of Data as they are dereferenced to different descriptions of the same real-world entity and thus allow different views and opinions to be expressed on the Web. In order to still be able to track that different information providers speak about the same entity, it is common practice that information providers set *owl:sameAs* links to URI aliases they know about.

While there has been some debate in the Semantic Web community regarding the use of *owl:sameAs* to link entities that some perceive not to be the same, it remains the case that *owl:sameAs* statements are simply claims that a data publisher makes about the world. If a data consumer disagrees with the veracity of a publisher’s claims they are free to not consume that dataset.

Different communities have specific preferences on the vocabularies they prefer to use for publishing data on the Web. The Web of Data is therefore open to arbitrary vocabularies being used in parallel. Despite this general openness, it is considered good practice to reuse terms from well-known RDF vocabularies such as FOAF, SIOC, SKOS, DOAP, vCard, Dublin Core, OAI-ORE, or GoodRelations wherever possible in order to make it easier for client applications to process linked data. Only if these vocabularies do not provide the required terms should data publishers define new, data source-specific terminology [29]. If new terminology is defined, it should be made self-describing by making the URIs that identify terms Web dereferencable [33]. This allows clients to retrieve RDF Schema or OWL definitions of the terms as well as term mappings to other vocabularies, such as *owl:equivalentClass*, *owl:equivalentProperty*, *rdfs:subClassOf*, or *rdfs:subPropertyOf* triples. The Web of Data thus relies on a pay-as-you-go data integration approach [34] based on a mixture of using common vocabularies together with data source-specific terms that are connected by mappings as deemed necessary.

A common serialization format for linked data is RDF/XML [35]. In situations where human inspection of RDF data is required, Notation3 [36], and its subset Turtle [37], are often provided as alternative, interconvertible serializations, due to the greater perceived readability of these formats. Alternatively, linked data can also be serialized as RDFa [38] which provides for embedding RDF triples into HTML. In the second case, data publishers should use the RDFa *about* attribute to assign URIs to entities, thereby enabling other data providers to reference these entities in other data sources external to the document. More details on RDFa can be found in [▶ Semantic Annotation and Retrieval: Web of Hypertext – RDFa and Microformats](#).

6.2.3 Publishing Tools

A variety of linked data publishing tools has been developed. The tools either serve the content of RDF stores as linked data on the Web or provide linked data views over non-RDF legacy data sources. The tools shield publishers from dealing with technical details such as content negotiation and ensure that data are published according to the linked data community best practices [14, 29, 32, 33]. All tools support dereferencing URIs into RDF descriptions. In addition, some of the tools also provide SPARQL query access to the served datasets and support the publication of RDF dumps.

- *D2R Server*. D2R Server [39] is a tool for publishing non-RDF relational databases as linked data on the Web. Using a declarative mapping language, the data publisher defines a mapping between the relational schema of the database and the target RDF vocabulary. Based on the mapping, D2R server publishes a linked data view over the database and allows clients to query the database via the SPARQL protocol. URI requests and SPARQL queries are rewritten into SQL queries against the underlying database, which ensures that generated RDF reflects the current state of the database.

- *Virtuoso Universal Server*. The OpenLink Virtuoso server (<http://www.openlinksw.com/dataspace/dav/wiki/Main/VOSRDF>) provides for serving RDF data via a linked data interface and a SPARQL endpoint. RDF data can either be stored directly in Virtuoso or can be created on-the-fly from non-RDF relational databases based on a mapping.
- *Talis Platform*. The Talis Platform (<http://www.talis.com/platform/>) is delivered as Software as a Service accessed over HTTP, and provides native storage for RDF/Linked Data. Access rights permitting, the contents of each Talis Platform store are accessible via a SPARQL endpoint and a series of REST APIs that adhere to the linked data principles.
- *Pubby*. The Pubby server [40] can be used as an extension to any RDF store that supports SPARQL. Pubby rewrites URI requests into SPARQL DESCRIBE queries against the underlying RDF store. Besides RDF, Pubby also provides a simple HTML view over the data store and takes care of handling 303 redirects and content negotiation between the two representations.
- *Triplify*. The Triplify toolkit [41] supports developers in extending existing Web applications with linked data front-ends. Based on SQL query templates, Triplify provides a linked data and as a JSON view over the application's database.
- *Drupal RDF CCK* module [42] enables site administrators to export their site content model and data to the Web of Data without requiring extensive knowledge on Semantic Web technologies. The module creates RDFa annotations and – optionally – a SPARQL endpoint for any Drupal site.
- *OAI2LOD Server*. The OAI2LOD [43] is a linked data wrapper for document servers that support the Open Archives OAI-RMH protocol.

A service that helps publishers to debug their linked data site is the Vapour validation service (<http://vapour.sourceforge.net/>). Vapour verifies that published data comply with the linked data principles and community best practices [14, 29, 32, 33].

6.2.4 Link Generation

RDF links allow client applications to navigate between data sources and to discover additional data. In order to be part of the Web of Data, data sources should set RDF links to related entities in other data sources. As data sources often provide information about large numbers of entities, it is common practice to use automated or semiautomated approaches to generate RDF links.

In various domains, there are generally accepted naming schemas. For instance, in the publication domain there are ISBN and ISSN numbers, in the financial domain there are ISIN identifiers, EAN and EPC codes are widely used to identify products, in life science various accepted identification schemas exist for genes, molecules, and chemical substances. If the link source and the link target datasets already both support one of these identification schema, the implicit relationship between entities in both datasets can easily

be made explicit as RDF links. This approach has been used to generate links between various data sources in the LOD cloud, for instance, ISBN numbers were used to set links between DBpedia and RDF book mash-up, CAS numbers were used to interlink Drugbank with Bio2RDF.

If no shared naming schema exist, RDF links are often generated based on the similarity of entities within both datasets. Such similarity computations can build on a large body of related work on record linkage [44] and duplicate detection [45] within the database community as well as on ontology matching [46] in the knowledge representation community. An example of a similarity-based interlinking algorithm is presented in [47]. In order to set RDF links between artists in the Jamendo and Musicbrainz datasets, the authors use a similarity metric that compares the names of artists as well as the titles of their albums and songs.

A new type of linked data tool that has recently emerged are link discovery frameworks, which automatically generate RDF links between data sources based on a matching description. An example of such a framework is Silk [48]. Using the declarative Silk-Link Specification Language (Silk-LSL), data publishers can specify which types of RDF links should be discovered between data sources as well as which conditions data items must fulfill in order to be interlinked. These link conditions can apply different similarity metrics to multiple properties of an entity or related entities which are addressed using a path-based selector language. The resulting similarity scores can be weighted and combined using various similarity aggregation functions. Silk works against local and remote SPARQL endpoints and is designed to be employed in distributed environments without having to replicate datasets locally.

In real-world settings, data are often not as clean and complete as they should be. For instance, two data sources might both support the same identification schema, like EAN, ISBN, or ISIN numbers, but due to a large number of missing values, it is nevertheless necessary to use similarity computations in addition to identifier matching to generate links. Such data quality problems are usually not known in advance but discovered when a data publisher tries to compute links pointing to a target data source. Therefore, finding a good linking heuristic is usually an iterative process. In order to support users with this task, Silk provides a Web interface for evaluating the correctness and completeness of generated links. Based on this evaluation, users can fine-tune their linking specification, for example, by changing weights or applying different metrics or aggregation functions. This Web interface is shown in [▶ Fig. 6.3](#). A second example, the LinQL framework [49], works over relational databases and is designed to be used together with database to RDF mapping tools such as D2R Server or Virtuoso.

6.2.5 Meta-Information About Published Data

Linked data should be published alongside with several types of meta-information in order to increase their utility for data consumers.

Silk:web

Compare Resource Pairs

Link Triples (.nb):

```
<http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugs/DB00814> <http://dbpedia.org/about/html/http://www.v3.org/2002/07/owl#sameAs>
<http://dbpedia.org/resource/Meloxicam> .
<http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugs/DB01036> <http://dbpedia.org/about/html/http://www.v3.org/2002/07/owl#sameAs>
<http://dbpedia.org/resource/Tolterodine> .
<http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugs/DB00378> <http://dbpedia.org/about/html/http://www.v3.org/2002/07/owl#sameAs>
<http://dbpedia.org/resource/Dydrogesterone> .
<http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugs/DB00948> <http://dbpedia.org/about/html/http://www.v3.org/2002/07/owl#sameAs>
<http://dbpedia.org/resource/Fulvestrant> .
<http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugs/DB01033> <http://dbpedia.org/about/html/http://www.v3.org/2002/07/owl#sameAs>
<http://dbpedia.org/resource/Metoclopramide> .
<http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugs/DB00025> <http://dbpedia.org/about/html/http://www.v3.org/2002/07/owl#sameAs>
<http://dbpedia.org/resource/Anakinra> .
```

LinkSpec ID: Reverse pair order High: 0.999999839972 Low: 0.224517813463 Average: 0.626288195802 Exceptions: 0

Source Resource	Target Resource	Similarity (click row for details)
http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugs/DB00378	http://dbpedia.org/resource/Dydrogesterone	0.9999998399723472
http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugs/DB01036	http://dbpedia.org/resource/Tolterodine	0.9999996927690115
http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugs/DB00814	http://dbpedia.org/resource/Meloxicam	0.9999985771322385
http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugs/DB00948	http://dbpedia.org/resource/Fulvestrant	0.2728216702567859
http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugs/DB00025	http://dbpedia.org/resource/Anakinra	0.26039158122051537
http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugs/DB01033	http://dbpedia.org/resource/Metoclopramide	0.22451781346294356

■ Fig. 6.3

Silk-Web interface for evaluating the quality of generated links in order to fine-tune the linking specification

6.2.5.1 Provenance Meta-Information

The quality of published data may vary widely. In order to enable clients to assess data quality and determine whether they want to trust data, data should be accompanied with meta-information about their creator, their creation date as well as the creation method [50]. Basic provenance meta-information can be provided using Dublin Core terms or the Semantic Web Publishing vocabulary [51]. The Open Provenance Model [52] provides terms for describing data transformation workflows. In [53], the authors propose a method for providing evidence for RDF links and for tracing how the RDF links change over time. If data publishers want to warrant provenance meta-information with digital signatures, they can rely on the method proposed in [51].

6.2.5.2 Technical Meta-Information

In order to support clients in choosing the most efficient way to access Web data for the specific task they have to perform, data publishers can provide additional technical meta-information about their dataset and its interlinkage relationships with other datasets. The Semantic Web Crawling sitemap extension [54] allows data publishers to state which alternative means of access (SPARQL endpoint, RDF dumps) are provided besides dereferencable URIs. The Vocabulary of Interlinked Datasets (void) [55] defines

terms and best practices to categorize and provide statistical meta-information about datasets as well as the link sets connecting them. By providing the terms with which to describe datasets, voidD also provides a basis for attaching licensing, waiver, and rights information to specific groupings of related data.

6.2.5.3 Licensing

Applications that consume data from numerous distributed and heterogeneous sources must be able to access explicit specifications of the terms under which data can be reused and republished, in the form of rights, waivers, and community norms. Availability of appropriate frameworks for publishing such specifications is an essential requirement in encouraging data owners to participate in the Web of Data, and in providing assurances to data consumers that they are not infringing the rights of others by using data in a certain way. Initiatives such as the Creative Commons (<http://creativecommons.org/>) have provided a framework for the open licensing of creative works, underpinned by the notion of copyright. However, as Miller et al. discuss in [56], copyright law is not applicable to factual data, which from a legal perspective is also treated differently across jurisdictions. Therefore, frameworks such as the Open Data Commons Public Domain Dedication and License (<http://www.opendatacommons.org/odc-public-domain-dedication-and-licence/>) or the Creative Commons Zero public domain dedication (<http://creativecommons.org/publicdomain/zero/1.0/>) should be adopted by the community to provide clarity in this area. In situations where attribution is a condition of data reuse, further research may also be required to explore how this can be achieved in user interfaces that combine data from large numbers of sources.

6.3 Example Applications: Retrieving and Consuming from the Web of Data

The unique characteristics and capabilities of linked data have implications for the kinds of applications that may be built on the Web of Data, and how they may be architected. This section will discuss some of these implications and review a number of Web of Data applications that have been deployed to date.

6.3.1 Application Architectures for the Web of Data

At present it is common for a particular dataset to be closely associated with a specific software application. For example, in most cases it is common to access a particular e-mail mailbox using just one or two applications (perhaps a desktop application and a Webmail client). The underlying content, that is, the e-mail messages, is effectively siloed and inaccessible to other applications. Consequently, the data contained in these e-mail messages, either explicitly or implicitly, are rarely reused automatically in other applications, for example, to populate a bookmarking application with links sent by e-mail.

The ease of reuse of linked data may lead to a trend where a particular dataset is less tightly coupled to a specific application, but instead is consumed on demand by various applications supporting different user tasks. While a specific application may remain the primary or only means to modify a specific dataset, adoption of linked data principles can enable this dataset to be reused in other applications that may themselves contribute additional linked data, thereby enhancing the original dataset. The logical extension of this trend is that data become the first-class citizen of the computing environment, with applications simply providing read/write views over one or more datasets.

From an application development perspective linked data have the following characteristics that influence the architecture of Web of Data applications:

1. Data are strictly separated from formatting and presentational aspects.
2. Data are self-describing. If an application consuming linked data encounters data described with an unfamiliar schema, the application can dereference the URI of the schema itself to find its definition.
3. The use of HTTP as a standardized data access mechanism and RDF as a standardized data model simplifies data discovery and access compared to Web APIs, which rely on heterogeneous data models and access interfaces.
4. Standardized data representation and access mechanisms facilitate the serendipitous discovery and reuse of data across applications, which may themselves contribute additional linked data that enhance the original dataset.

Given these factors, a number of architectural considerations must be taken into account when developing applications for the Web of Data, particularly related to data retrieval, integration, and prioritization.

Where applications are consuming data from numerous heterogeneous sources that may or may not be known in advance, efficient procedures must be in place for the retrieval of data in a timely fashion. This may be enabled through advance crawling and caching, or on-the-fly at application runtime through link traversal or federated querying. Search engines such as Sindice [57], Falcons [58], and Watson [59] (more on Watson can be found in [▶ Semantic Web Search Engines](#)) crawl the Web of Data and provide applications with access to crawled data through APIs. Federated query architectures for linked data include DARQ [60] and SemaPlorer [61]. The Semantic Web Client Library [62] has demonstrated that expressive queries can be answered against the Web of Data by relying on runtime link traversal.

The appropriate mixture of these methods will always depend on the specific needs of a linked data application. However, due to the likelihood of latency and query optimization problems with on-the-fly link traversal and federated querying, it may transpire that widespread crawling and caching will become the norm in making data available to applications in a timely fashion, while being able to take advantage of the openness of the Web of Data by discovering new data sources through link traversal. This may require the development of a new generation of services that extend the model adopted by indexes such as Sindice and Watson, by providing sophisticated query access over sets of linked data assembled from the Web.

On-the-fly retrieval through link traversal may be a suitable approach for building applications that are able to preemptively retrieve related data in the background, while users are engaged in a particular task for which data are already available. For example, if an application can cache data that are frequently reused across a wide range of tasks, it may be possible to retrieve additional task-specific data as the needs of the user become more apparent or specific. However, experience with Web of Data browsers presented below, suggests that retrieval times for data are too great to make this approach feasible without more sophisticated heuristics for predicting the data requirements of a user at a particular time.

The majority of Web of Data applications developed to date can be broadly classified into three categories: Web of Data browsers, Web of Data search engines, and Web of Data mash-ups. The following section will examine each of these categories.

6.3.2 Web of Data Browsers

Just as traditional Web browsers allow users to navigate between HTML pages by following hypertext links, linked data browsers allow users to navigate between data sources by following links expressed as RDF triples. For example, a user may view DBpedia's RDF description of the city of Birmingham (UK), follow a "birthplace" link to the description of the comedian Tony Hancock (who was born in the city), and from there onward into RDF data from the BBC describing broadcasts in which Hancock starred. The result is that a user may begin navigation in one data source and progressively traverse the Web by following RDF rather than HTML links. The Disco hyperdata browser (<http://www4.wiwiw.fu-berlin.de/bizer/ng4j/disco/>) follows this approach and can be seen as a direct application of the hypertext navigation paradigm to the Web of Data.

Access to linked data, however, provides human interface opportunities and challenges beyond those of the hypertext Web. People need to be able to explore the Web of links between items, but also to powerfully analyze data in bulk. The Tabulator [63, 64], for example, allows the user to traverse the Web of Data, and expose pieces of it in a controlled way, in "outline mode"; to discover and highlight a pattern of interest; and then query for any other similar patterns in the data Web. The results of the query form a table that can then be analyzed with various conventional data presentation methods, such as faceted browsers, maps, and timelines.

In contrast, while authors such as those of [65] have questioned the use of graph-oriented views over RDF data, Hastrup, Cyganiak, and Bojars argue in [66] that such interfaces fill an important niche, and describe their Fenfire browser that follows such a paradigm.

Tabulator and Marbles [67] (see [▶ Fig. 6.4](#) below) are among the browsers that track data provenance, while merging data about the same entity from different sources based on owl:sameAs links and Inverse Functional Properties. This feature provides an integrated view of data about a specific entity, while still allowing the user to determine the source of particular fragments of the data.

The screenshot shows the Marbles browser interface. At the top, the address bar contains the URL `http://www.w3.org/People/Berners-Lee/card#i` and an 'Open' button. The main content area displays the profile for 'Tim Berners-Lee'. The profile includes a header with the name and a 'marbles' logo. Below the header, there are several rows of data, each with a label on the left and a list of values on the right. The values are often accompanied by small colored circles representing different types or sources of data. A central image of Tim Berners-Lee is displayed. The data rows include:

- label:** Tim Berners-Lee
- nameAs:** Tim Berners-Lee (also at `www.w3.org`)
- image:** A photograph of Tim Berners-Lee sitting at a desk.
- Weblinks:** `http://www.w3.org/People/Berners-Lee/`
- name:** Tim Berners-Lee, Timothy Berners-Lee, Tim Berners-Lee
- Given name:** Timothy
- family name:** Berners-Lee
- sha1sum of a personal mailbox URI name:** `965d47da70b7407210ae0ef4e95374a025dc6`
- workspace homepage:** `http://www.w3.org/`
- nickname:** TimBL, TimBL, Embi
- personal mailbox:** `mailto:timbl@w3.org`
- seeAlso:** Tim Berners-Lee's FOAF file, Tim Berners-Lee's FOAF file
- is seeAlso of:** Tim Berners-Lee, Tim Berners-Lee
- special:** `http://www.w3.org/People/Berners-Lee/`
- is primary topic of:** `http://en.wikipedia.org/wiki/Tim_Berners-Lee`
- knows:** Coralia Menier

Fig. 6.4

The Marbles browser displaying data about Tim Berners-Lee

6.3.3 Web of Data Search Engines and Indexes

In the traditional hypertext Web, browsing and searching are often seen as the two dominant modes of interaction [68]. While browsers provide the mechanisms for navigating the information space, search engines are often the place at which that navigation process begins. A number of search engines have been developed that crawl linked data from the Web by following RDF links, and provide query capabilities over aggregated data. Broadly speaking, these services can be divided into two categories: human-oriented search engines, and application-oriented indexes.

6.3.3.1 Human-Oriented Search Engines

Search engines such as Falcons [58] and SWSE [69] provide keyword-based search services oriented toward human users, and follow a similar interaction paradigm to existing market leaders such as Google and Yahoo!. The user is presented with a search

box into which they can enter keywords related to the item or topic in which they are interested, and the application returns a list of results that may be relevant to the query. However, rather than simply providing links from search results through to the source documents in which the queried keywords are mentioned, both SWSE and Falcons provide a more detailed interface to the user that exploits the underlying structure of the data. Both provide a summary of the entity the user selects from the results list, alongside additional structured data crawled from the Web and links to related entities.

It is interesting to note that while both SWSE and Falcons operate over corpuses of structured data crawled from the Web, and exploit this structure when presenting search results, they choose to provide very simple query capabilities that mimic the query interfaces of conventional Web search engines. While one may intuitively expect the additional structure in the data to be exploited to provide sophisticated query capabilities for advanced users, this has not proved to be the case to date, with the exception of Tabulator's style of query-by-example and faceted browsing interfaces for query refinement. SWSE does provide access to its underlying data store via the SPARQL query language; however, this is suitable primarily for application developers with a knowledge of the language rather than regular users wishing to ask very specific questions through a usable human interface.

Falcons provides users with the option of searching for objects, concepts, and documents, each of which leads to slightly different presentation of results. While the object search (► [Fig. 6.5](#)) is suited to searching for people, places, and other more concrete items, the concept search is oriented to locating classes and properties in ontologies published on the Web. The document search feature provides a more traditional search engine experience, where results point to RDF documents that contain the specified search terms.

Where data are available about the location of an object, Falcons also displays this on a map. In fact, in the case of these search engines that provide entity-centric interfaces to items featured in search results, the conceptual distinction between Web of Data search engines and the Web of Data browsers described above is minimal, with the exception of whether data are cached in advance or accessed on-the-fly. Rather than supporting wholly distinct modes of interaction, these two styles of application differ primarily in whether or not they provide an entity lookup feature with which to start a browsing process, or simply an "address bar" in which to enter the URI for a particular entity. This form of convergence between application types previously considered distinct may represent a key influence of the Web of Data on future directions in human-computer interaction. One application that demonstrates such convergence is SemaPlorer [61], which combines search functionality with varied browsing mechanisms such as facets, maps, and timeline views.

One final comparison worth making is between human-oriented search engines that operate primarily over the Web of Data, and those that exploit structured data published within conventional HTML documents on the Web. For example, through their SearchMonkey (<http://developer.yahoo.com/searchmonkey/>) product, Yahoo! have begun to exploit structured data published in HTML pages as RDFa or Microformats, and use these to provide richer and more structured results to users. Similar initiatives

The screenshot shows the Falcons search interface. At the top, there are tabs for 'Object', 'Concept', and 'Document'. A search box contains the text 'Berlin' and a 'Search Objects' button. Below the search box, a note says 'Separate keywords with a space, and put a phrase in double quotes.' A section titled 'Specify a type:' contains a grid of category buttons: Agent, Event, Motion Picture Film, Album, Facility, Organization (highlighted), Building, Group (highlighted), Person, City, Landmark, State, Concept, Location, and Subject. Below this is a header for search results: 'Objects 1 - 10 of 42,186 for your search Berlin (2.4 seconds)'. Three results are shown, each with a title, a list of properties, and a URL. The first result is 'Berlin is a State, Capital, City' with properties like abstract, subject, and photo collection. The second is '_Berlin is a Thing, _Category-3AStadt' with properties like article, defined by, and label. The third is 'Berlin is a Thing, Subject, City' with properties like defined by, page, and label.

Object [Concept](#) [Document](#)

Berlin

Separate keywords with a space, and put a phrase in double quotes.

Specify a type:

Agent	Album	Building	City	Concept
Event	Facility	Group	Landmark	Location
Motion Picture Film	Organization	Person	State	Subject

Objects 1 - 10 of 42,186 for your search Berlin (2.4 seconds)

Berlin is a *State, Capital, City*

- abstract: **Berlin** redirige para aqui. Para otros significados, v... - [From dbpedia.org »](#)
- has subject: *Category:13th_century_establishments* - [From dbpedia.org »](#)
- hasPhotoCollection: **Berlin** - [From dbpedia.org »](#)

<http://dbpedia.org/resource/Berlin> - Described in 1855 documents

_Berlin is a *Thing, _Category-3AStadt*

- hasArticle: **Berlin** - [From www.sembase.at »](#)
- isDefinedBy: <http://www.sembase.at/index.php/Special:ExportRDF/Berlin> - [From www.sembase.at »](#)
- label: **Berlin** - [From www.sembase.at »](#)

http://wiki.sembase.at/index.php/_Berlin - Described in 17 documents

Berlin is a *Thing, Subject, City*

- isDefinedBy: <http://ontoworld.org/wiki/Special:ExportRDF/Berlin> - [From ontoworld.org »](#)
- page: **Berlin** - [From ontoworld.org »](#)
- label: **Berlin** - [From ontoworld.org »](#)

<http://ontoworld.org/wiki/Special:URIResolver/Berlin> - Described in 16 documents

■ Fig. 6.5

Falcons object search results for the keyword “Berlin”

have been announced by Google, in the form of their Rich Snippets (<http://googlewebmastercentral.blogspot.com/2009/05/introducing-rich-snippets.html>) program. For a full discussion of RDFa we refer the reader to [► Semantic Annotation and Retrieval: Web of Hypertext – RDFa and Microformats](#), this volume.

While both approaches address the same goal of a richer information-seeking experience for the user, some distinctions remain. For example, while RDFa represents simply another serialization of RDF and as such can be used to publish RDF links, at present, the Yahoo! crawler only traverses the Web of documents by following HTML links, in order to build its index. In contrast, the Web of Data search engines discussed above primarily traverse linked data by following RDF links. Similarly, while Web of Data search engines such as SWSE and Falcons provide entity-centric views of results and some degree of data integration, Yahoo!’s search results remain document-oriented, meaning that the same (redundant) data about a certain entity can appear alongside multiple entries in search results if they are published in multiple locations on the Web. Over time this distinction is likely to disappear, from both a user interaction and a technical infrastructure perspective.

6.3.3.2 Application-Oriented Indexes

While SWSE and Falcons provide search capabilities oriented toward humans, another breed of services have been developed to serve the needs of applications built on top of

distributed linked data. These application-oriented indexes, such as Swoogle [70], Sindice [57], and Watson [59], provide APIs through which linked data applications can discover RDF documents on the Web that reference a certain URI or contain certain keywords. The rationale for such services is that each new linked data application should not need to implement its own infrastructure for crawling and indexing all parts of the Web of Data of which it might wish to make use. Instead, applications can query these indexes to receive pointers to potentially relevant documents, which can then be retrieved and processed by the application itself.

Despite this common theme, these services have slightly different emphases. Sindice is oriented more toward providing access to documents containing instance data (in formal terms the ABox), although this focus reflects as much the goals of the developers as the capabilities of the application. In contrast, the emphasis of Swoogle and Watson is on finding ontologies (i.e., the TBox) that provide coverage of certain concepts relevant to a query. We refer the reader to [▶ Semantic Web Search Engines](#), this volume for a comprehensive discussion on Search Engines in a Semantic Web context.

6.3.4 Web of Data Mash-ups

While the Web of Data browsers and search engines described above provide largely generic functionality, a number of services have been developed that offer more domain-specific functionality by “mashing up” data from various linked data sources. Not only do such mash-ups exploit existing links between data sources, but where they are able to make new connections between related entities these can also be published back into the Web in RDF, further increasing the link density of the Web and avoiding duplication of data integration efforts by other data consumers.

6.3.4.1 Revyu

Revyu [22] is a generic reviewing and rating site that follows a similar model to existing sites of this nature, such as epinions.com, but is based on linked data principles and the Semantic Web technology stack. In addition to publishing data according to the techniques and best practices described above, Revyu consumes linked data from the Web to enhance the experience of site users. For example, when films are reviewed on Revyu, the site attempts to match these with the corresponding entry in DBpedia. Where a match is made, additional information about the film (such as the director’s name and the film poster) is retrieved from DBpedia and shown in the human-oriented (HTML) pages of the site. In addition, links are made at the RDF level to the corresponding item and are published in the machine-oriented RDF document describing the film. This approach ensures that while human users see a richer view of the item through the mashing up of data from various sources, linked data-aware applications are simply provided with

references to URIs that identify the same item in other datasets, and from which related data may be retrieved.

Similar principles are followed for linking items such as books and pubs to their corresponding entries in external datasets such as the RDF Book Mashup and the Open Guides [24], while users of the site can choose to provide the URI of a FOAF profile, which is used to enhance their Revyu user profile without requiring this information to be duplicated in multiple locations. These techniques not only reduce data redundancy across sites, but minimize the amount of domain-specific data that Revyu must hold in order to provide a richer user experience.

6.3.4.2 DBpedia Mobile

DBpedia Mobile [67] is a location-aware linked data browser designed to be run on an iPhone or other mobile device. Based on the current GPS position of a mobile device, DBpedia Mobile renders a map indicating nearby locations from the DBpedia dataset (► Fig. 6.6). Starting from this map, the user can explore background information about his surroundings by navigating along data links into other Linking Open Data sources. While sightseeing provides the initial use case for the application, it is not restricted to



► Fig. 6.6

DBpedia Mobile displaying data from DBpedia and Revyu about the Brandenburg Gate in Berlin

a fixed set of data sources but can retrieve and display data from arbitrary Web data sources. Consequently, DBpedia Mobile can also be employed in other use cases as a generic Web of Data mash-up application. Besides accessing Web data, DBpedia Mobile also enables users to publish their current location, pictures, and reviews to the Web of Data so that they can be used by other Semantic Web applications. Instead of simply being tagged with geographical coordinates, published content is interlinked with a nearby DBpedia resource and thus contributes to the overall richness of the Web of Data.


6.3.4.3 Talis Aspire

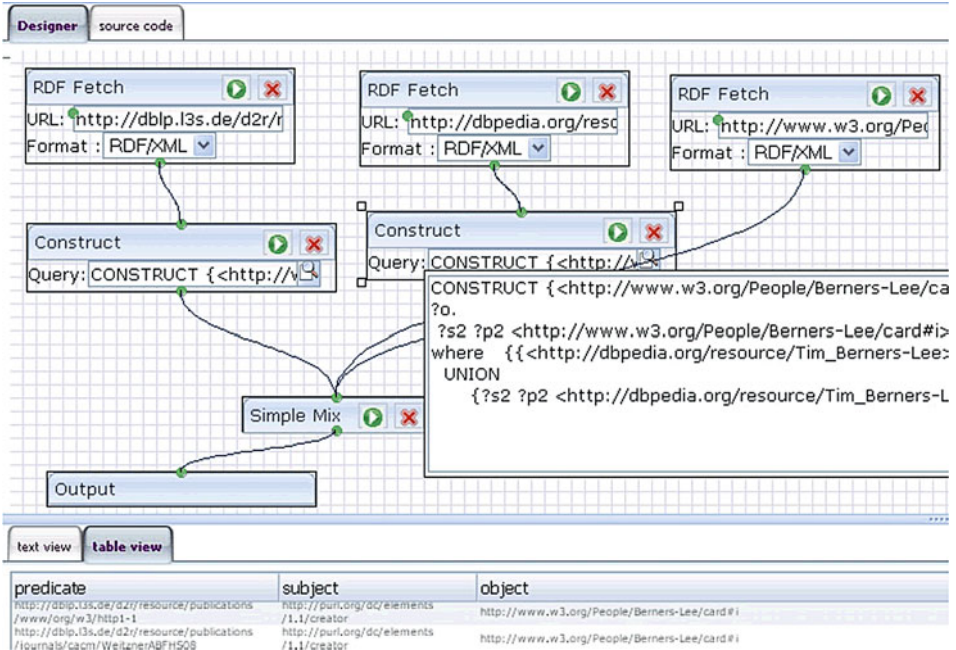
Talis Aspire [71] is a Web-based Resource List Management application deployed to university lecturers and students. As users create lists through a conventional Web interface, the application produces RDF triples which are persisted to an underlying linked data-compatible store. The use of linked data principles enables items present on one list to be transparently linked to the corresponding items featured on lists at other institutions, thereby building a Web of scholarly data through the actions of nonspecialist users.

6.3.4.4 BBC Programs and Music

The British Broadcasting Corporation (BBC) uses linked data internally as a lightweight data integration technology. The BBC runs numerous radio stations and television channels. Traditionally, these stations and channels use separate content management systems. The BBC has thus started to use linked data technologies together with DBpedia and MusicBrainz as controlled vocabularies to connect content about the same topic residing in different repositories and to augment content with additional data from the Linking Open Data cloud. Based on these connections, BBC Programmes and BBC Music build linked data sites for all of its music and programs related brands [19].

6.3.4.5 DERI Pipes

Perhaps the most generic application to date offering mash-up capabilities over linked data are DERI Pipes [72]. Modeled on Yahoo Pipes, DERI Pipes provides a data-level mash-up platform that enables data sources to be plugged together to form new feeds of data. The resulting aggregation workflows may contain sophisticated operations such as identifier consolidation, schema mapping, RDFS or OWL reasoning, with data transformations being expressed using SPARQL CONSTRUCT operations or XSLT templates.  *Figure 6.7* shows the assembly of a workflow to integrate data about Tim Berners-Lee within the DERI pipes development environment.



■ Fig. 6.7

DERI pipes workflow integrating data about Tim Berners-Lee from three data sources

6.4 Related Resources

This section lists key papers that laid the foundation for the Web of Data or have significantly influenced its development. Afterward, community websites and events are listed.

6.4.1 Key Papers

- Berners-Lee, T.: *Linked Data – Design Issues* [14]. An initial Web design note in which Tim Berners-Lee outlines the linked data principles and lays the foundations for the Web of Data.
- Bizer, C., Heath, T., Berners-Lee, T.: *Linked Data – The Story So Far* [6]. Overview article that explains the linked data principles, the progress in publishing linked data on the Web, reviews applications that have been developed to exploit this ecosystem, and maps out a research agenda for the Web of Data to move forward.
- Berners-Lee, T et al.: *Tabulator: Exploring and Analyzing linked data on the Semantic Web* [63]. A paper about Tabulator, the first linked data browser developed at MIT, which outlines the foundations for applications to access the Web of Data.

- Bizer, C., Cyganiak, R., Heath, T.: *How to publish Linked Data on the Web* [29]. A tutorial covering various aspects of linked data publication and providing concrete recipes for serving linked data on the Web.
- Sauermann, L., Cyganiak, R.: *Cool URIs for the Semantic Web* [32]. A W3C Interest Group Note presenting guidelines on how to use URI references for identifying arbitrary entities and on how to serve data describing these entities on the Web using two alternative strategies called *303 URIs* and *hash URIs*.
- Diego Berrueta, Jon Phipps: *Best Practice Recipes for Publishing RDF Vocabularies* [33]. W3C Working Group Note describing best practice recipes for publishing vocabularies or ontologies on the Web (in RDF Schema or OWL). Each recipe introduces general principles and an example configuration for use with an Apache HTTP server. The recipes are all designed to be consistent with the architecture of the Web as currently specified.
- Jacobs, I., Walsh, N.: *Architecture of the World Wide Web, Volume One* [2]. A W3C Recommendation specifying the general architecture of the World Wide Web and thus laying the foundation for linked data which directly applies this architecture to publishing data on the Web.

6.4.2 Community Websites

- *Linked Data – Connect Distributed Data across the Web* (<http://linkeddata.org/>). A community website providing a home for, or pointers to, resources from across the linked data community such as tutorials, presentations, tools, datasets, calls for papers and events.
- *W3C Linking Open Data* community effort pages in the W3C ESW wiki (<http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>). A website of the W3C Linking Open Data community effort, listing community news, papers, presentations, community and scientific events. The main page contains the current version of the LOD cloud diagram visualizing the Web of Data. Subpages provide listings of datasets published by the community, RDFizers, publishing tools, linked data browsers and search engines, as well as link discovery tools.

6.4.3 Linked Data Events

- *Linked Data on the Web (LDOW)* workshop series at World Wide Web conferences. Started in 2008, the LDOW workshop series provides a forum for presenting the latest research on linked data and drives forward the research agenda in this area. While the LDOW2008 workshop in Beijing focused on the publication of linked data, the LDOW2009 workshop in Madrid focused on linked data application architectures, linking algorithms and Web data fusion.

- *Linked Data Meets Artificial Intelligence (LinkedAI2010)* workshop at AAAI Spring Symposium, March 2010, Stanford, USA. A symposium aiming at bringing together the researchers working on linked data and AI and to create a new community interested in utilizing AI techniques such as ontologies, machine learning, data fusion, etc. in exploring the Web of Data.
- *VoCamps* event series (http://vocamp.org/wiki/Main_Page). A series of informal, grassroots events where people spend some dedicated time creating lightweight vocabularies/ontologies for the Web of Data. The emphasis of the events is not on creating the perfect ontology in a particular domain, but on creating vocabularies that are good enough for people to start using for publishing data on the Web. Up till now, 12 VoCamps have taken place around the globe in cities such as Oxford, Bristol, Galway, Sunnyvale, New York City, Seoul, and Washington DC.
- *LOD Community Gatherings*. A series of informal meetings of the W3C Linking Open Data community usually taking place in conjunction with the World Wide Web conference and the International Semantic Web conference. Up till now, 12 LOD Community Gatherings have taken place around the globe.

6.5 Future Issues

By publishing and interlinking various data sources on the Web, the Linking Open Data community has created a crystallization point for the Web of Data and a challenging test bed for linked data technologies. However, to address the ultimate goal of being able to use the Web as a single global database, various remaining research challenges must be overcome.

6.5.1 User Interfaces and Interaction Paradigms

Arguably, the key benefit of linked data from the user perspective is the provision of integrated access to data from a wide range of distributed and heterogeneous data sources, on a scale that has not been readily feasible previously. By definition, this may involve integration of data from sources not explicitly selected by users, as to do so would likely incur an unacceptable cognitive overhead. While the Web of Data browsers described above demonstrate promising trends in how applications may be developed that exploit linked data, numerous challenges remain in understanding appropriate user interaction paradigms for applications built on data assembled dynamically in this fashion.

A number of these challenges are highlighted by Heath in [73]. For example, while hypertext browsers provide mechanisms for navigation forward and backward in a document-centric information space, similar navigation controls in a linked data browser should enable the user to move forward and backward between entities rather than documents, changing the focal point of the application. Linked data browsers will also need to provide intuitive and effective mechanisms for adding and removing data

sources from an integrated, entity-centric view. Sigma [74], a search engine based on the Sindice service, gives an indication of how such functionality could be delivered. However, understanding how such an interface can be realized when data sources number in the thousands or millions is a captivating research challenge.

As discussed above, a shift in interaction paradigm from document-centricity to entity-centricity blurs the distinction between browsers and search engines. Rather than maintaining this distinction, efforts must be made to understand, at a more fundamental level, the goals of users in interacting with the Web as a data space. Numerous attempts have been made to identify specific forms of the search or browse tasks on the document Web [75]. However, it has been argued in [76] that these studies are prone to conflating the goals of the user with meta-tasks or artifacts of the current generation of applications. An entity-centric Web may enable applications that natively support a far broader range of tasks than can be described by variations on searching and browsing, such as monitoring or sharing a resource. If the full potential of the Web of Data from a user perspective is to be realized, the community must engage in fundamental research aimed at understanding these tasks.

One additional issue concerns how data publishers can support data consumers in visualizing data that they have made available. This may be particularly significant in cases where data are published according to vocabularies that an application has not previously encountered. One approach in this direction is the Fresnel display vocabulary [77], which enables schema and data providers to publish generic lenses which provide hints to client applications on how the data should be visualized.

6.5.2 Schema Mapping and Data Fusion

Once data have been retrieved from distributed sources, it must be integrated in a meaningful way before it is displayed to the user or is further processed. Today, most linked data applications display data from different sources alongside each other but do little to integrate it further. To do so does requires a mapping of terms from different vocabularies to the applications target schema, as well as fusing data about the same entity from different sources, by resolving data conflicts.

Linked data sources either use their own schemas or use a mixture of terms from existing, well-known vocabularies together with self-defined terms specific to the particular data source. In order to support clients in transforming data between different schemas, data sources can publish correspondences between their local terminology and the terminology of related data sources on the Web of Data. Current W3C recommendations such as RDF Schema [30] and OWL [31] define basic terminology like *owl:equivalentClass*, *owl:equivalentProperty*, *rdfs:subClassOf*, *rdfs:subPropertyOf* that can be used to publish basic correspondences. In many situations, these correspondences are too coarse-grained to properly transform data between schemas. Problems include, for instance, structural heterogeneity as well as value transformations. An open research issue is therefore the development of languages to publish finer-grained schema mappings on

the Web. Ideally, such languages would support transitive mappings and provide for combining partial mappings in order to cover cases where data sources mix terminology from different vocabularies. Candidate technologies for this include the alignment languages presented in [78] and [79] as well as the rules interchange format (RIF) (http://www.w3.org/2005/rules/wiki/RIF_Working_Group).

In addition to enhanced support for schema mapping, further research is needed in the area of data fusion for linked data applications. Data fusion is the process of integrating multiple data items representing the same real-world object into a single, consistent, and clean representation. The main challenge in data fusion is the resolution of data conflicts, that is, choosing a value in situations where multiple sources provide different values for the same property of an object. There is a large body of work on data fusion in the database community [80] and an increasing body of work on identity reconciliation in the Web community [81]. Specific requirements that distinguish the Web of Data from other data fusion scenarios arise from the autonomy of data sources and the scarceness and uncertainty of quality-related meta-information that is required to assess data quality in order to resolve inconsistencies. Prototypical systems for fusing linked data from multiple sources include DERI Pipes [72] and the KnoFuss architecture [82].

6.5.3 Trust, Quality, and Relevance

A significant consideration for linked data applications is how to ensure the data most relevant or appropriate to the user's needs are identified and made available. For example, in scenarios where data quality and trustworthiness are paramount, how can this be determined heuristically, particularly where the dataset may not have been encountered previously?

An overview of different content-, context-, and rating-based techniques that can be used to heuristically assess the relevance, quality, and trustworthiness of data is given in [83, 84]. Equivalents to the PageRank algorithm will likely be important in determining coarse-grained measures of the popularity or significance of a particular data source, as a proxy for relevance or quality of the data; however, such algorithms will need to be adapted to the linkage patterns that emerge on the Web of Data.

From an interface perspective, the question of how to represent the provenance and trustworthiness of data drawn from many sources into an integrated view is a significant research challenge. Berners-Lee proposed in [85] that browser interfaces should be enhanced with an "Oh, yeah?" button to support the user in assessing the reliability of information encountered on the Web. Whenever a user encounters a piece of information that they would like to verify, pressing such a button would produce an explanation of the trustworthiness of the displayed information. This goal has yet to be realized; however, existing developments such as WIQA [83] and InferenceWeb [86] can contribute to work in this area by providing explanations about information quality as well as inference processes that are used to derive query results.

6.5.4 Link Maintenance

The content of linked data sources changes: data about new entities are added; outdated data are changed or removed. Today, RDF links between data sources are updated only sporadically, which leads to dead links pointing at URIs that are no longer maintained and to potential links not being set as new data is published. The Web architecture is in principle tolerant to dead links, but having too many of them leads to a large number of unnecessary HTTP requests by client applications. A current research topic within the linked data community is therefore link maintenance. Proposed approaches to this problem range from recalculating links at regular intervals using frameworks such as Silk [48] or LinQL [49], through data sources publishing update feeds [41], or informing link sources about changes via subscription models to central registries such as Ping the Semantic Web that keep track of new or changed data items.

6.5.5 Privacy

The ultimate goal of linked data is to be able to use the Web like a single global database. The realization of this vision would provide benefits in many areas but will also aggravate dangers in others. One problematic issue is the opportunity to violate privacy that arises from integrating data from distinct sources, particularly where reasoning is used to infer additional data, or where third parties can release data about an individual, thereby infringing his or her privacy. Protecting privacy in the Web of Data context is likely to require a combination of technical and legal means together with a higher awareness of the users about what data to provide in which context. As Shabajee reports in [87], maintaining this awareness can be challenging even for technically astute users. Interesting research initiatives in this domain are Weitzner's work on the privacy paradox [88] and the recent work by the TAMI project on information accountability [89].

References

1. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.* **30**(1–7), 107–117 (1998)
2. Jacobs, I., Walsh, N.: Architecture of the world wide web, volume one, W3C Recommendation. <http://www.w3.org/TR/webarch/> (2004). Accessed 14 June 2009
3. Berners-Lee, T., et al.: Uniform Resource Identifier (URI): generic syntax. Request for comments: 3986. <http://tools.ietf.org/html/rfc3986> (2005). Accessed 14 June 2009
4. Fielding, R., et al.: Hypertext Transfer Protocol – HTTP/1.1. Request for comments: 2616. <http://www.w3.org/Protocols/rfc2616/rfc2616.html> (1999). Accessed 14 June 2009
5. Klyne, G., Carroll, J.: Resource description framework (RDF): concepts and abstract syntax, W3C Recommendation. <http://www.w3.org/TR/rdf-concepts/> (2004). Accessed 14 June 2009
6. Bizer, C., Heath, T., Berners-Lee, T.: Linked data – the story so far. *Int. J. Semantic Web Inf. Syst.* **5**(3), 1–22 (2009)
7. Bizer, C.: The emerging web of linked data. *IEEE Intell. Syst.* **24**(5), 87–92 (2009)
8. Berners-Lee, T., et al.: The world-wide web. *Commun. ACM* **37**(8), 76–82 (1994)

9. Berners-Lee, T.: Weaving the web. Harper, San Francisco (1999)
10. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. *Sci. Am.* **284**(5), 34–43 (2001)
11. Marshall, C., Shipman, F.: Which semantic web? In: Proceedings of the 14th ACM Conference on Hypertext and Hypermedia (HT 2003), Nottingham (2003)
12. Fielding, R.: Architectural styles and the design of network-based software architectures. Ph.D. dissertation, University of California, Irvine (2000)
13. Franklin, M., Halevy, A., Maier, D.: From databases to dataspace: a new abstraction for information management. *ACM SIGMOD Rec.* **34**(4), 27–33 (2005)
14. Berners-Lee, T.: Linked data – design issues. <http://www.w3.org/DesignIssues/LinkedData.html> (2006). Accessed 23 July 2010
15. Hausenblas, M., Halb, W., Raimond, Y., Heath, T.: What is the size of the semantic web? In: Proceedings of the International Conference on Semantic Systems (I-Semantics 2008), Graz (2008)
16. Bizer, C., Cyganiak, R., Gauß, T.: The RDF book mashup: from web APIs to a web of data. In: Proceedings of the Third Workshop on Scripting for the Semantic Web (SFSW 2007), Innsbruck (2007)
17. Van de Sompel, H., Lagoze, C., Nelson, M., Warner, S., Sanderson, R., Johnston, P.: Adding e-Science assets to the data web. In: Proceedings of the Second Workshop on Linked Data on the Web (LDOW 2009), Madrid (2009)
18. Hassanzadeh, O., Consens, M.: Linked movie data base. In: Proceedings of the Second Workshop on Linked Data on the Web (LDOW 2009), Madrid (2009)
19. Kobilarov, G., et al.: Media meets semantic web – how the BBC uses DBpedia and linked data to make connections. In: Proceedings of the Sixth European Semantic Web Conference (ESWC 2009), Heraklion. Lecture Notes in Computer Science, vol. 5554, pp. 723–737. Springer, Berlin (2009)
20. Belleau, F., Nolin, M., Tourigny, N., Rigault, P., Morissette, J.: Bio2RDF: towards a mashup to build bioinformatics knowledge systems. *J. Biomed. Inform.* **41**(5), 706–716 (2008)
21. Jentsch, A., Hassanzadeh, O., Bizer, C., Andersson, B., Stephens, S.: Enabling tailored therapeutics with linked data. In: Proceedings of the Second Workshop on Linked Data on the Web (LDOW 2009), Madrid (2009)
22. Heath, T., Motta, E.: Revyu: Linking reviews and ratings into the Web of data. *J. Web Semant.* **6**(4), 266–273 (2008)
23. Auer, S., Lehmann, J., Hellmann, S.: LinkedGeoData – adding a spatial dimension to the web of data. In: Proceedings of the Eighth International Semantic Web Conference (ISWC 2009), Chantilly. Lecture Notes in Computer Science, vol. 5823, pp. 731–746. Springer, Berlin (2009)
24. Gaved, M., Heath, T., Eisenstadt, M.: Wikis of locality: insights from the open guides. In: Proceedings of the 2006 International Symposium on Wikis (WikiSym 2006), Odense (2006)
25. Bizer, C., et al.: DBpedia – a crystallization point for the web of data. *J. Web Semant. Sci.* **7**, 154–165 (2009)
26. Berners-Lee, T.: Putting government data online, web architecture design note. <http://www.w3.org/DesignIssues/GovData.html> (2009). Accessed 23 July 2010
27. Birbeck, M.: RDFa and linked data in UK government websites. *Nodalities Magazine*, Aug–Sept 2009, 15–16
28. Acar, S., Alonso, J., Novak, K.: Improving access to government through better use of the web, W3C Interest Group Note. <http://www.w3.org/TR/egov-improving/> (2009). Accessed 23 July 2010
29. Bizer, C., Cyganiak, R., Heath, T.: How to publish linked data on the web. <http://www4.wiwiwiss.fu-berlin.de/bizer/pub/LinkedDataTutorial/> (2007). Accessed 14 June 2009
30. Brickley, D., Guha, R.: RDF vocabulary description language 1.0: RDF schema, W3C Recommendation. <http://www.w3.org/TR/rdf-schema/> (2004). Accessed 14 June 2009
31. McGuinness, D., van Harmelen, F.: OWL Web ontology language, W3C Recommendation. <http://www.w3.org/TR/owl-features/> (2004). Accessed 14 June 2009
32. Saueremann, L., Cyganiak, R.: Cool URIs for the semantic web, W3C Interest Group Note. <http://www.w3.org/TR/cooluris/> (2008). Accessed 14 June 2009
33. Berrueta, D., Phipps, J.: Best practice recipes for publishing RDF vocabularies, W3C Working Group Note. <http://www.w3.org/TR/swbp-vocab-pub/> (2008). Accessed 14 June 2009

34. Das Sarma, A., Dong, X., Halevy, A.: Bootstrapping pay-as-you-go data integration systems. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2008), Vancouver (2008)
35. Beckett, D.: RDF/XML syntax specification (revised), W3C Recommendation. <http://www.w3.org/TR/rdf-syntax-grammar/> (2004). Accessed 14 June 2009
36. Berners-Lee, T.: Notation3 (N3) a readable RDF syntax. <http://www.w3.org/DesignIssues/Notation3.html> (1998). Accessed 23 July 2009
37. Beckett, D., Berners-Lee, T.: Turtle – terse RDF triple language, W3C Team Submission. <http://www.w3.org/TeamSubmission/turtle/> (2008). Accessed 23 July 2009
38. Adida, B., et al.: RDFa in XHTML: syntax and processing, W3C Recommendation. <http://www.w3.org/TR/rdfa-syntax/> (2008). Accessed 14 June 2009
39. Bizer, C., Cyganiak, R.: D2R server – publishing relational databases on the semantic web. In: Poster at the Fifth International Semantic Web Conference (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273. Springer, Berlin (2006)
40. Cyganiak, R., Bizer, C.: Pubby – a linked data frontend for SPARQL endpoints. <http://www4.wiwiw.fu-berlin.de/pubby/> (2008). Accessed 14 June 2009
41. Auer, S., et al.: Triplify – light-weight linked data publication from relational databases. In: Proceedings of the 18th World Wide Web Conference (WWW 2009), Madrid (2009)
42. Corlosquet, S., Delbru, R., Clark, T., Polleres, A., Decker, S.: Produce and consume linked data with Drupal! In: Proceedings of the Eighth International Semantic Web Conference (ISWC 2009). Chantilly. Lecture Notes in Computer Science, vol. 5823, pp. 763–778. Springer, Berlin (2009)
43. Haslhofer, B., Schandl, B.: The OAI2LOD server: exposing OAI-PMH metadata as linked data. In: Proceedings of the First Workshop about Linked Data on the Web (LDOW 2008), Beijing (2008)
44. Winkler, W.: Overview of record linkage and current research directions. US Bureau of the Census, Technical report, Washington, DC (2006)
45. Elmagarmid, A., Ipeirotis, P., Verykios, V.: Duplicate record detection: a survey. *IEEE Trans. Knowl. Data Eng.* **19**(1), 1–16 (2007)
46. Euzenat, J., Shvaiko, P.: *Ontology Matching*. Springer, Heidelberg (2007)
47. Raimond, Y., Sutton, C., Sandler, M.: Automatic interlinking of music datasets on the semantic web. In: Proceedings of the First Workshop About Linked Data on the Web (LDOW 2008), Beijing (2008)
48. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Discovering and maintaining links on the web of data. In: Proceedings of the Eighth International Semantic Web Conference (ISWC 2009), Chantilly. Lecture Notes in Computer Science, vol. 5823, pp. 650–665. Springer, Berlin (2009)
49. Hassanzadeh, O., et al.: (2009). A declarative framework for semantic link discovery over relational data. In: Poster at 18th World Wide Web Conference (WWW 2009), Madrid (2009)
50. Hartig, O.: Provenance information in the web of data. In: Proceedings of the Second Workshop on Linked Data on the Web (LDOW 2009), Madrid (2009)
51. Carroll, J., Bizer, C., Hayes, P., Stickler, P.: Named graphs. *J. Web Semant.* **3**(4), 247–267 (2005)
52. Moreau, L., et al.: The open provenance model. Technical report, Electronics and Computer Science, University of Southampton Southampton (2008)
53. Zhao, J., Klyne, G., Shotton, D.: Provenance and linked data in biological data webs. In: Proceedings of the First Workshop about Linked Data on the Web (LDOW 2008), Beijing (2008)
54. Cyganiak, R., Delbru, R., Stenzhorn, H., Tummarello, G., Decker, S.: Semantic sitemaps: efficient and flexible access to datasets on the semantic web. In: Proceedings of the Fifth European Semantic Web Conference (ESWC 2008), Tenerife. Lecture Notes in Computer Science, vol. 5021, pp. 690–704. Springer, Berlin (2008)
55. Alexander, K., Cyganiak, R., Hausenblas, M., Zhao, J.: Describing linked datasets. In: Proceedings of the Second Workshop on Linked Data on the Web (LDOW 2009), Madrid (2009)
56. Miller, P., Styles, R., Heath, T.: Open data commons, a license for open data. In: Proceedings of the First Workshop about Linked Data on the Web (LDOW 2008), Beijing (2008)
57. Oren, E., et al.: Sindice.com: a document-oriented lookup index for open linked data. *Int. J. Metadata Semant Ontol.* **3**(1), 37–52 (2008)

58. Cheng, G., Qu, Y.: Searching linked objects with Falcons: approach, implementation and evaluation. *Int. J. Semant. Web Inf. Syst.* 5(3), (2009)
59. D'Aquin, M., et al.: Toward a new generation of semantic web applications. *IEEE Intell. Syst.* 23(3), 20–28 (2008)
60. Quilitz, B., Leser, U.: Querying distributed RDF data sources with SPARQL. In: *Proceedings of the Fifth European Semantic Web Conference (ESWC 2008)*, Tenerife. *Lecture Notes in Computer Science*, vol. 5021, pp. 524–538. Springer, Berlin (2008)
61. Schenk, S., et al.: SemaPlorer – interactive semantic exploration of data and media based on a federated cloud infrastructure. In: *Proceedings of the Semantic Web Challenge at ISWC 2008*, Karlsruhe (2008)
62. Hartig, O., Bizer, C., Freytag, J.C.: Executing SPARQL queries over the web of linked data. In: *Proceedings of the Eighth International Semantic Web Conference (ISWC 2009)*, Chantilly. *Lecture Notes in Computer Science*, vol. 5823, pp. 293–309. Springer, Berlin (2009)
63. Berners-Lee, T., et al.: Tabulator: exploring and analyzing linked data on the semantic web. In: *Proceedings of the Third International Semantic Web User Interaction Workshop (SWUI 2006)*, Athens (2006)
64. Berners-Lee, T., et al.: Tabulator redux: browsing and writing linked data. In: *Proceedings of the First Workshop on Linked Data on the Web (LDOW 2008)*, Beijing (2008)
65. Karger, D., Schraefel, M.C.: Pathetic fallacy of RDF. In: *Proceedings of Third Semantic Web User Interaction Workshop (SWUI 2006)*, Athens (2006)
66. Hastrup, T., Cyganiak, R., Bojars, U.: Browsing linked data with Fenfire. In: *Proceedings of the First Workshop About Linked Data on the Web (LDOW 2008)*, Beijing (2008)
67. Becker, C., Bizer, C.: Exploring the geospatial semantic web with DBpedia mobile. *J. Web Semant.* 7, 278–286 (2009)
68. Olston, C., Chi, E.: ScentTrails: integrating browsing and searching on the web. *ACM Trans. Comput. Hum. Interact.* 10(3), 177–197 (2003)
69. Hogan, A., Harth, A., Umrich, J., Decker, S.: Towards a scalable search and query engine for the web. In: *Proceedings of the 16th Conference on World Wide Web (WWW 2007)*, Banff (2007)
70. Ding, L., et al.: Finding and ranking knowledge on the semantic web. In: *Proceedings of the Fourth International Semantic Web Conference (ISWC 2005)*, Galway. *Lecture Notes in Computer Science*, vol. 3729, pp. 156–170. Springer, Berlin (2005)
71. Clarke, C.: A resource list management tool for undergraduate students based on linked open data principles. In: *Proceedings of the Sixth European Semantic Web Conference (ESWC 2009)*, Heraklion. *Lecture Notes in Computer Science*, vol. 5554, pp. 697–707. Springer, Berlin (2009)
72. Le Phuoc, D., Polleres, A., Morbidoni, C., Hauswirth, M., Tummarello, G.: Rapid semantic web mashup development through semantic web pipes. In: *Proceedings of the 18th World Wide Web Conference (WWW 2009)*, Madrid (2009)
73. Heath, T.: How will we interact with the web of data? *IEEE Internet Comput.* 12(5), 88–91 (2008)
74. Catasta, M., Cyganiak, R., Tummarello, G.: Towards ECSSE: live web of data search and integration. In: *Proceedings of the Semantic Search 2009 Workshop (SemSearch 2009) at WWW 2009*, Madrid (2009)
75. Rose, D. E., Levinson, D.: Understanding user goals in web search. In: *Proceedings of the 13th International Conference on World Wide Web (WWW 2004)*, New York (2004)
76. Heath, T., Dzbor, M., Motta, E.: Supporting user tasks and context: challenges for semantic web research. In: *Proceedings of the Workshop on End-User Aspects of the Semantic Web (UserSWeb)*, European Semantic Web Conference (ESWC 2005), Heraklion (2005)
77. Pietriga, E., Bizer, C., Karger, D., Lee, R.: Fresnel – a browser-independent presentation vocabulary for RDF. In: *Proceedings of the Fifth International Semantic Web Conference (ISWC 2006)*, Athens, GA. *Lecture Notes in Computer Science*, vol. 4273, pp. 158–171. Springer, Berlin (2006)
78. Haslhofer, B.: A Web-based mapping technique for establishing metadata interoperability. Ph.D. thesis, Universität Wien, Vienna (2008)
79. Euzenat, J., Scharffe, F., Zimmermann, A.: Expressive alignment language and implementation. *Knowledge web project report, KWEB/2004/D2.2.10/1.0* (2007)
80. Bleiholder, J., Naumann, F.: Data fusion. *ACM Comput. Surv.* 41(1), 1–41 (2008)

81. Halpin, H., Thomson, H.: Special issue on identity, reference and the web. *Int. J. Semant. Web. Inf. Syst.* **4**(2), 1–72 (2008)
82. Nikolov, A., et al.: Integration of semantically annotated data by the KnoFuss architecture. In: *Proceedings of the 16th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2008)*, Acitrezza. *Lecture Notes in Computer Science*, vol. 5268. Springer, Berlin (2008)
83. Bizer, C., Cyganiak, R.: Quality-driven information filtering using the WIQA policy framework. *J. Web Semant.* **7**(1), 1–10 (2009)
84. Heath, T.: Information-seeking on the web with trusted social networks – from theory to systems. Ph.D. thesis, The Open University, Milton Keynes (2008)
85. Berners-Lee, T.: Cleaning up the user interface, section – the “Oh, yeah?”-button. <http://www.w3.org/DesignIssues/UI.html> (1997). Accessed 14 June 2009
86. McGuinness, D., da Silva, P.: Infrastructure for web explanations. In: *Proceedings of the Second International Semantic Web Conference (ISWC 2003)*, Sanibel Island. *Lecture Notes in Computer Science*, vol. 2870, pp. 113–129. Springer, Berlin (2003)
87. Shabajee, P.: Informed consent on the semantic web – issues for interaction and interface designers. In: *Proceedings of the Third International Semantic Web User Interaction Workshop (SWUI 2006)*, *Fifth International Semantic Web Conference (ISWC 2006)*, Athens, GA (2006)
88. Weitzner, D.: Beyond secrecy: new privacy protection strategies for open information spaces. *IEEE Internet Comput.* **11**(5), 94–96 (2007)
89. Weitzner, D., et al.: Information accountability. *Commun. ACM* **51**(6), 82–87 (2008)



7 Storing the Semantic Web: Repositories

Atanas Kiryakov · Mariana Damova

Ontotext AD, Sofia, Bulgaria

7.1	<i>Introduction</i>	233
7.1.1	Inspiring Vision + Standards + Business Demands = Rapid Development	235
7.1.2	Semantic Repositories = Inference Engines + Column-Stores	236
7.1.3	RDF, SPARQL, and RDF-Based Data Representation Models	240
7.1.4	Linked Data	244
7.2	<i>Reasoning in the Semantic Repositories</i>	245
7.2.1	Lightweight Inference Integration	247
7.2.1.1	Rule-Based Entailment Example and Relations to Query Evaluation	248
7.2.1.2	Reasoning Strategies: Forward- and Backward-Chaining	249
7.2.1.3	The Advantages and Applicability of the Different Strategies	250
7.2.1.4	Hybrid Strategies and Dynamic Materialization	251
7.2.1.5	The Honey and the Sting of owl:sameAs	252
7.2.1.6	Truth Maintenance and Smooth Invalidation	253
7.2.2	OWL Dialects Suitable for Scalable Inference	254
7.3	<i>Semantic Repository Tasks, Performance Factors, and Dimensions</i>	258
7.3.1	Tasks	258
7.3.2	Performance Factors	259
7.3.3	Performance Dimensions	259
7.3.4	Full-Cycle Benchmarking	260
7.4	<i>Performance Considerations and Distribution Approaches</i>	261
7.4.1	Performance Engineering and Hardware Trends	261
7.4.1.1	Usage Scenario: 1 Million Queries per Day Against 10 Billion Statements	262
7.4.1.2	The \$10,000 Database Server Landmark	262
7.4.2	Distributed Semantic Repositories	263
7.4.2.1	Data Partitioning	264
7.4.2.2	Data Replication	265
7.4.2.3	Advantages of the Different Distribution Approaches	266
7.4.3	Multi-Core Parallelism	266

7.5	<i>Popular Benchmarks and Datasets</i>	267
7.5.1	Lehigh University Benchmark (LUBM) and UOBM	267
7.5.2	UniProt	268
7.5.3	DBPedia and Other Linked Datasets	269
7.5.4	Berlin SPARQL Benchmark (BSBM)	269
7.6	<i>Semantic Repository Engines</i>	270
7.6.1	3store, 4store, 5store	270
7.6.2	AllegroGraph	271
7.6.3	BigData	271
7.6.4	BigOWLIM	272
7.6.5	DAML DB, Asio Parliament	273
7.6.6	Jena TDB	273
7.6.7	ORACLE	274
7.6.8	Sesame	275
7.6.9	Virtuoso	275
7.7	<i>State of the Art in Performance and Scalability</i>	277
7.7.1	Data Loading Performance	277
7.7.2	Query Evaluation	278
7.7.2.1	BSBM Results	279
7.7.2.2	LUBM Results	280
7.8	<i>Typical Applications</i>	282
7.8.1	Reason-Able Views to the Web of Linked Data	282
7.8.1.1	FactForge: The Upper-Level Knowledge Base	284
7.8.1.2	Linkedlifedata: 25 Biomedical Databases in a Box	286
7.8.2	Publishing of Content on the Web, Based on Semantic Metadata	288
7.9	<i>Related Resources</i>	290
7.10	<i>Future Issues</i>	293
7.11	<i>Cross-References</i>	294

Abstract: Semantic repositories are database management systems, capable of handling structured data, taking into consideration their semantics. The Semantic Web represents the next-generation Web of Data, where information is published and interlinked in a way, which facilitates both humans and machines to exploit its structure and meaning. To foster the realization of the Semantic Web, the World Wide Web Consortium (W3C) developed a series of metadata, ontology, and query languages for it. Following the enthusiasm about the Semantic Web and the wide adoption of the related standards, today, most of the semantic repositories are database engines, which deal with data represented in RDF, support SPARQL queries, and can interpret schemas and ontologies represented in RDFS and OWL. Naturally, such engines take the role of Web servers of the Semantic Web.

This chapter starts with an introduction to semantic repositories and discussion on their links to several other technology trends, including relational databases, column-stores, and expert systems. As the most distinguishing quality of the semantic repositories is reasoning, an overview of the strategies for the integration of inference in the data management life cycle is presented. An overall view of the mechanics of the engines is provided from the perspective of a conceptual framework that reveals all their tasks and activities (e.g., storage and retrieval) along with the factors that impact their performance (e.g., data size and complexity). A review of several design issues, including distribution, serves as a basis for understanding the different implementation approaches and their implications on the performance of semantic repositories. Several of the most popular benchmarks and datasets, which are often used as measuring sticks for the performance of the engines, and few of the outstanding semantic repositories, are presented along with the best published evaluation results.

The advantages and the typical applications of semantic repositories are presented focusing on two usage scenarios: reasoning with and the management of linked data (a popular trend in the Semantic Web) and enterprise data integration. The chapter ends with some considerations regarding the future development of semantic repositories and design topics like adaptive indexing and interoperability patterns.

7.1 Introduction

The Semantic Web creates a wealth of data, where information is given well-defined meaning and computers are better able to work with it; a vast collection of structured data are published and interlinked together to form a Web of Data. The potential for increasing knowledge availability and the ability of machines to effectively work with it is enormous. Managing the data on the Web, however, represents a tremendous challenge, considering their size and complexity, the anticipated number of requests, and the desirable response times. This opens the story of semantic repositories (SR) – tools that combine characteristics of database management systems (DBMS) and inference engines to support efficient manipulation of Semantic Web data. Semantic repositories take the role of Web servers, providing access to the Web of Data.

The presence of such data management systems, able to hold, interpret, and serve requests and queries from multiple users against massive amounts of data, is an indispensable step toward the realization of the vision and the potential of the Semantic Web. They evolve dynamically, racing to extend human's and computer's capabilities to deal with structured data. The implementation of such engines requires advancement of the frontiers in two fields: databases and reasoning. Each new development allows loading more data, dealing with more comprehensive schemas and ontologies, and answering more complex queries in less time. As in mountain climbing, each new achievement uncovers new opportunities and challenges.

A story related to UNIPROT illustrates what it feels like to be a technology pioneer. UNIPROT is the most extensive and the most popular public database about protein-related information (see [▶ Sect. 7.5.2](#)). Back in 2006, Ontotext established LifeSKIM – a small team to work on life science applications. One of the first tasks of the LifeSKIM people was to load UNIPROT in the OWLIM semantic repository (see [▶ Sect. 7.6.4](#)). The first attempt ended up with a “stack overflow” error – the corresponding version of the engine was not prepared for a group of 3,000 concepts, related through the transitive `owl:sameAs` property. Once this problem was fixed, it became clear that heavy usage of `owl:sameAs` alignment can ruin the performance, as discussed in [▶ Sect. 7.2.1.4](#). The necessary optimizations in the indices were made; latter on those proved to be very useful for linked data management and data integration (see [▶ Sect. 7.8.1](#)). It was finally possible to load UNIPROT and to perform materialization against the relevant fragment of OWL. The last surprise came when a person from the LifeSKIM team defined several sample queries and tried to make sense of the results. Some of them were definitely incorrect, and investigation was carried out to find the source of the problem. Finally, it appeared that there was a small bug in the UNIPROT schema, which remained unspotted by its developers and its numerous users, because no one before was able to interpret this aspect of its semantics.

The UNIPROT encounter with OWLIM (the semantic repository) is an example of how a dataset can drive an improvement in the engine and vice versa. Practically, semantic repositories can be seen as track-laying machines, which extend the reach of the data railways: Each previous step is only possible on top of the results of the previous one. These railways change the data-economy of entire domains and areas, by allowing larger volumes of more complex data to be handled at lower cost. This makes the topics around performance, capabilities, and development of semantic repositories both fascinating and intriguing.

This chapter deals with the foundations of semantic repositories and attempts to provide a roadmap toward their major characteristics, related design and performance issues, the state of the art in the field, and future directions. The major objectives are:

- To clarify the principles of operation of semantic repositories and the benefits of their usage
- To explain the facets of their performance, because their understanding of this is a key factor for the successful adoption of semantic repositories

The remainder of this section starts with discussion on the recent history and the “political economy” of the field ([▶ Sect. 7.1.1](#)), the role of the semantic repositories and

their typical usage (➤ Sect. 7.1.2), and continues with a quick introduction to several related subjects: RDF data models (➤ Sect. 7.1.3) and linked data basics (➤ Sect. 7.1.4).

➤ Section 7.2 discusses reasoning within the semantic repositories: the strategies for the integration of inference in the data management life cycle, with their advantages and related problems, as well as, ontology languages and dialects suitable for inference in such scenarios. In ➤ Sect. 7.3, a conceptual framework for the understanding of all tasks and activities of a semantic repository is provided. This framework addresses also the factors that impact its performance and the different aspects or dimensions of this performance. ➤ Section 7.4 goes into a range of practical issues related to the design of today's semantic repositories, including analysis of typical server configurations and various approaches for the distribution of the repositories. Next, in ➤ Sect. 7.5, several of the most popular benchmarks and datasets are presented – those are often used as measuring sticks for the performance of the semantic repository engines. At this point, the floor is set to discuss a few of the most prominent semantic repositories (➤ Sect. 7.6) and present an overview of the best published evaluation results (➤ Sect. 7.7).

➤ Section 7.8 outlines three typical applications of semantic repositories:

- FactForge: a search engine, serving as a gateway, facilitating the usage of the central datasets in the Linking Open Data cloud
- LinkedLifeData: a platform for semantic data integration in life sciences domain
- BBC's website for World Cup 2010, which demonstrates how semantic technologies can enable optimizations in the publishing process

Finally, a list of related resources is provided (➤ Sect. 7.9) and discussion on a few of the key issues related to the future development of the semantic repositories (➤ Sect. 7.10), along with design considerations (like adaptive indexing) and interoperability patterns, which are likely to be adopted in order to resolve some of the bottlenecks of today's semantic technology applications.

7.1.1 Inspiring Vision + Standards + Business Demands = Rapid Development

Semantic repositories are still in the initial phase of rapid upward development. Since 2004, every couple of years, the engines have been getting an order of magnitude faster and more scalable. Such development has been enabled by several factors:

- Standards – standardization efforts related to the Semantic Web, most notably RDF(S), OWL, and SPARQL, provided a solid ground for development and good minimal levels of interoperability.
- Benchmarks – the performance of a semantic repository is a multidimensional phenomenon; the performance with respect to different tasks depends on a range of factors and parameters. Making sustainable progress in the performance of the engines requires adequate measuring sticks and methods, namely, benchmarks and evaluation

methodologies. They can provide clear indication about the cost, the applicability, and the benefits of each new optimization, approach, or technique with respect to the different aspects of the performance.

- Hardware – while a \$10,000 server was needed to load one billion statements in 2006, three years later, this task was achievable on a \$2,000 workstation. Running semantic repositories on commodity hardware comparable with the environments on which relational databases are run is considered as a serious asset.
- Performance engineering – the understanding about the optimal server configurations for different tasks, types of data, and query loads is far better today than 5 years ago.
- Data integration demands – globalization and the consolidation of the business increased the demand for data integration approaches, which scale efficiently to tens and hundreds of data sources. In the life sciences, the availability of hundreds of public databases, efficient access to which can facilitate medical research and drug development, served as a huge stimulus for the adoption of semantic repositories.
- Linked data enthusiasm – the emergence of the Web of linked data (see [Sect. 7.1.4](#)) provoked interest in industry (and many governments) to use public data and to publish databases in RDF, according to the linked data principles [5].

In summary, the growing demand for the management of heterogeneous and dynamic structured data met with a technology stream that already offers robust tools. The latter are backed by a vision, community, and standards that ensure its steady development. The overheads related to the adoption of higher-level data management paradigm became bearable in the light of the increasing hardware capabilities.

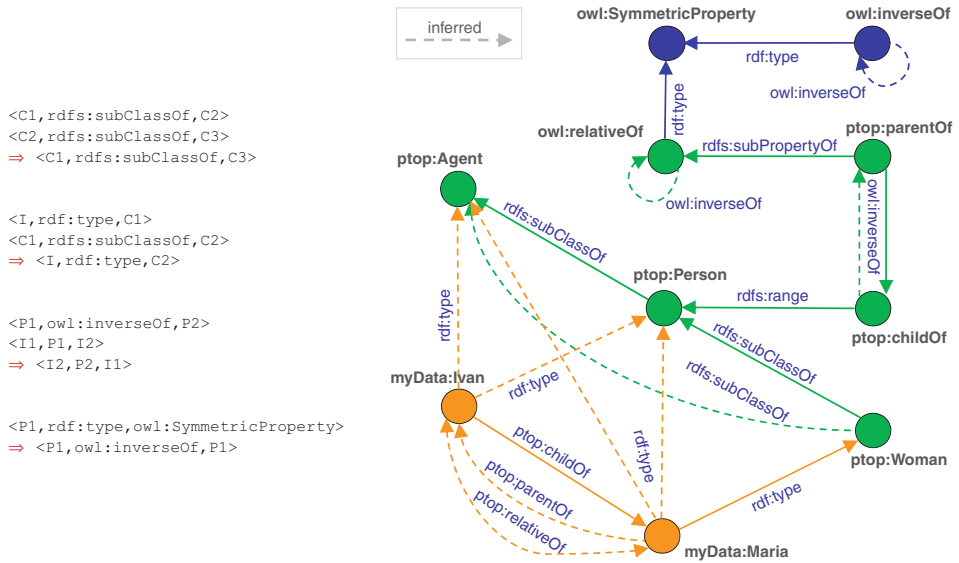
7.1.2 Semantic Repositories = Inference Engines + Column-Stores

Semantic repositories are engines similar to database management systems (DBMS). Their major functionality is to support efficient storage, querying, and management of structured data. The major differences with the DBMS can be summarized as follows:

- They use ontologies as semantic schemas, which allows them to automatically reason about the data.
- They work with generic physical data models, which allows them to easily adopt updates and extensions in the schemas, that is, in the structure of the data.

Functionally, semantic repositories are essentially DBMS that can interpret the data. Based on the semantics of the schemas, they can infer implicit facts and consider them in the process of query evaluation.

As an illustration of this, [Fig. 7.1](#) provides an example of the representation of simple family relationships (the graph on the right) and the facts that can be inferred from those (indicated with dashed arcs) based on simple entailment rules (presented on the left). Two explicit facts were asserted in the repository (the solid orange arcs): “Ivan is

**Fig. 7.1****Data representation and interpretation in semantic repositories**

a child of Maria” and “Maria is a woman.” The repository was able to infer several new facts (the dashed orange arcs), for example, that Maria is parent of Ivan, that they are relatives to each other, that Maria is a person and an agent. Those inferences were made on the basis of the specific data schema that was used. The interpretation of both the specific instance data and the schema was possible based on the semantics of the system primitives used (the part of the graph in blue), encoded in the entailment rules (presented on the left).

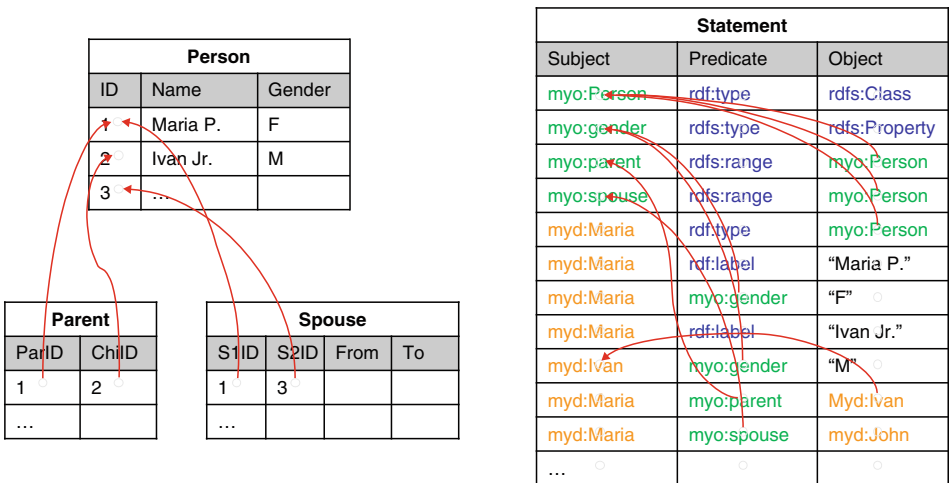
The most direct benefit of the ability of the semantic repository to interpret the data is that it can evaluate queries (or more generally retrieval requests) in a much “cleverer” and more flexible manner. For instance, in the above example, the repository will be able to return Ivan as a result of a request asking for all relatives of Maria (e.g., by the retrieval pattern “*Maria relativeOf ?x*”). In this case, the query pattern is more general than the explicit fact asserted (being a relative is a general case of being a child). Moreover, the relation in the query goes in the inverse direction (the fact was “*Ivan ?p Maria,*” but it matches request for “*Maria ?p ?x*”).

The above query example is an illustration that semantic repositories allow query variation with respect to the level of generality and the direction of the relation between two entities. More generally, the expression of the information need in the query no longer has to match the syntax of assertion of the data. This enables a whole new range of information access scenarios, where the person who formulates the query is not aware of the details of the schemas that were used for the encoding of the data. This is only possible in a DBMS where the engine understands (to some extent) the semantics of the data and the semantics of the query and can interpret them in order to find matches

despite syntax variations. This is probably the major difference between the RDF-based semantic repositories and XML-based DBMS. XML is designed to allow interoperability with respect to the syntax of the data, remaining ignorant to its semantics. For instance, in an XML schema, one cannot define inverse, transitive, and symmetric properties as it is possible with RDF (see [▶ Fig. 7.1](#)). This means that the XML database has no chance to interpret the data and to provide a useful answer to the query in the example given above.

As already mentioned, another principal advantage of the RDF-based DBMS is that they use a generic physical model. The data are represented internally in graph-like data structures like the one depicted on [▶ Fig. 7.1](#). [▶ Figure 7.2](#), on the other hand, presents a comparison of the representation of the data in relational DBMS (RDBMS, on the left) and the RDF databases (on the right), where each arc from the RDF graph is represented as a triple: subject (the source node), predicate (the property, which determines the type of relation or the attribute), and object (the target node, or the value). A change in the schema (e.g., definition of a new property) requires no changes in the representation of the asserted facts in the RDF database, because it does not impose a structural change in the physical representation. In contrast, in the relational DBMS, the physical representation of the data schema is dependent on the schema. The data are stored in files, structured in accordance with the structure of the tables, essentially as sequences of bytes of equal length, each of which representing one row in the table as depicted on [▶ Fig. 7.2](#) (on the left). Thus, a change in the schema (e.g., addition of a new column in a table) requires considerable re-arrangement of the data files.

Historically, over the last couple of decades, there were multiple attempts to implement “semantic databases” based on data understanding and interpretation. Those were labeled and promoted in very different ways, depending on their origin and the IT trends



■ Fig. 7.2

Row stores versus RDF databases

at the time of their appearance. The most notable language for “semantic databases” in this line is probably known as Datalog – a limited version of Prolog, defined for usage in DBMS (such databases were referred to as *deductive databases*). Most of the scalable semantic repositories today support logical languages quite similar in spirit and expressive power to Datalog (see [▶ Sect. 7.2.2](#)). Further, many of the expert systems and knowledge base management systems developed in the late 1980s and the early 1990s, as part of the Artificial Intelligence (AI) work, were also offering fairly similar functionality, packaged and promoted with plenty of high-level claims. Finally, over the last couple of decades, there have always been *reasoners* or *inference engines*. While these tools are mostly focused on logical inference, at the end of the day, their basic functionality is the same: One should be able to assert some facts and get answers based on interpretation of their semantics.

Over the last decade, the popularity of *column-stores* has been growing. The central idea is that, while in the relational database the information is stored and managed primarily by rows (those could be referred to as row-stores), in the column-stores, information is managed by columns. Typical representatives are Google’s BigTable, [12], and the Vertica Database, [66]. The major advantage of this representation is the same as with the RDF databases: Changes in the schema are far easier to implement compared to RDBMS. Such representations are also far more efficient for the management of sparse data. Imagine a class of objects that can have 50 different attributes. In relational databases, the information about the instances of such class will be naturally modeled as a table of 51 columns (one for primary key and 50 for the attributes). Now, imagine that for each instance of the class there are values defined only for 10 of these attributes on average. In a typical RDBMS, this would result in a data file, which is 80% full with null values. In contrast, an RDF-based DBMS does not need to allocate space for missing attribute values – there will be simply no such records in the Statements table as shown in the example on [▶ Fig. 7.2](#).

In a nutshell, column-stores have considerable advantages against the RDBMS in two respects: dynamic data schema and sparse data. The principal advantage of the RDBMS is that the information is already grouped to a major extent, so, typically, the RDBMS needs to make less joins (i.e., to resolve less references to other records) during query evaluation. Obviously, both column- and row-stores have their advantages in different data management scenarios. Column-stores are good in situations where aggregates are computed over large number of similar data items, like in data warehouses. Row-stores are good in situations where relatively stable processes with predetermined structure are to be automated and managed.

While RDF databases and column-stores share a lot of design principles, a typical column-store differs from an RDF-based semantic repository in several ways:

- *Globally unique identifiers*. An important feature of RDF, as a data representation model, is that it is based on the notion of Unique Resource Identifiers (URI, [4]). All predicates and most of the subjects (see [▶ Sect. 7.1.3](#)) are identified by a URI. In the examples provided above (see [▶ Fig. 7.2](#)), the nodes are identified by URLs

(`myd:maria` is a URL representation as QName, using namespace prefix, e.g., `myd:`). In contrast, most of the other DBMS use integer identifiers, which are unique only in the scope of the same type of elements in the same instance of the database.

- *Standard compliance.* While there are no well-established standards in the area of the column-stores, the RDF-based semantic repositories are highly interoperable between one another on the basis of a whole ecosystem of languages for schema definition, ontology definition, and querying.

Semantic repositories can be described as “*RDF-based column-stores with inference capabilities.*”

As a wrap-up, provided is a list of the major characteristics and advantages of semantic repositories:

- *Easy integration of multiple data sources:* Once the schemas of these sources are semantically aligned, the inference capabilities of the engine support the interlinking and combination of the facts from the different sources.
- *Easy querying against rich, diverse, or unknown data schemas:* Inference is applied to match the semantics of the query to the semantics of the data, regardless of the vocabulary and the data modeling patterns used for encoding data.
- *Great analytical power:* One can count that semantics will be thoroughly applied even when this requires recursive inferences on multiple steps. In this way, semantic repositories can uncover facts, based on the interlinking of long chains of evidence – the vast majority of those facts would remain unspotted in a regular DBMS.
- *Efficient data interoperability:* Importing RDF data from one store to another is straightforward, based on the usage of globally unique identifiers.

The above qualities make semantic repositories an attractive choice for a range of data management tasks, that is, data integration, data warehousing, content management, metadata management, master data management (MDM), online analytical processing (OLAP), and business intelligence (BI). Over the last couple of years, these applications have been recognized and analyzed by many reputable analysts of information technology (see [52] and [67]).

7.1.3 RDF, SPARQL, and RDF-Based Data Representation Models

Resource Description Framework (RDF) is a language for representing information about resources in the World Wide Web [41]. Although it was designed to represent metadata about Web resources, RDF has much broader use as a generic data model for structured data management and reasoning. SPARQL, [53], is a query language for RDF data sources. Here we provide an overview of several augmentations of the basic RDF specification,

which are relevant to its usage as data representation model in semantic repositories and the support of SPARQL queries.

The atomic data element in RDF represents a statement about a resource or a blank node. Each statement is a triple of the format $\langle \text{Subject}, \text{Predicate}, \text{Object} \rangle$, for example, $\langle \text{John}, \text{loves}, \text{Mary} \rangle$ or $\langle \text{Mary}, \text{hasBirthday}, "14.11.1972" \rangle$. RDF description can be seen as directed labeled graph, where each triple defines an edge directed from the subject to the object, which is labeled with the predicate. The nodes of the graph could be URI (unified resource identifiers, [4], e.g., an URL), blank node (auxiliary nodes), or XML literal. The predicates are always URIs. Literals are not allowed in subject position, that is, they cannot be the start of an edge in the graph. Intuitively, literals are used to describe resources identified by URIs, but there is no point in describing literals, because they represent primitive data values. A sample graph, which describes a Web page, created by a person called Adam, can be seen in [Fig. 7.3](#). More on RDF can be found in [Semantic Annotation and Retrieval: RDF](#).

SPARQL [53] is an SQL-like query language for RDF data, specified by the RDF Data Access Working Group of W3C. It differs from SQL in the following aspects:

- SPARQL does not contain specific Data Definition Language (DDL) provisions because the schemas are represented in both RDFS and OWL as standard RDF graphs, thus requiring no specific language to deal with them.
- SPARQL is not a Data Modification Language (DML), that is, one cannot insert, delete, and update RDF graphs using SPARQL. The major reason for this is that there is still no consensus on the optimal DML design for RDF.

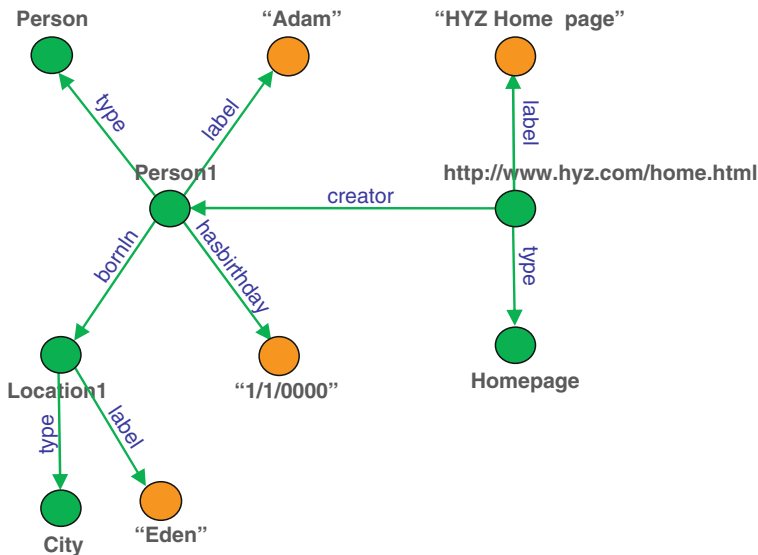


Fig. 7.3

RDF graph describing Adam and his home page

SPARQL supports four types of queries:

- SELECT queries – return n -tuples of results just like the SELECT queries in SQL.
- DESCRIBE queries – return an RDF graph. The resulting graph describes the resources, which match the query constraints. Usually, a description of a resource is considered an RDF-molecule, forming the immediate neighborhood of an URI.
- ASK queries – provide positive or negative answer indicating whether or not the query pattern can be satisfied.
- CONSTRUCT queries – return an RDF graph constructed by means of the substitution of the variables in the graph template and combining the triples into a single RDF graph by set union. More on SPARQL can be found in [Querying the Semantic Web: SPARQL](#).

Named graph, [11], is an RDF graph with assigned name in the form of a URI reference. In an extended RDF model, one can deal with multiple named graphs and describe the graphs, making statements about them, putting their URIs in subject position. While the original definition of named graphs leaves plenty of room for interpretation, a more concrete definition is provided in the specification of SPARQL, where queries are evaluated against datasets, composed from multiple RDF graphs. In SPARQL, [53], *RDF Dataset* is defined as

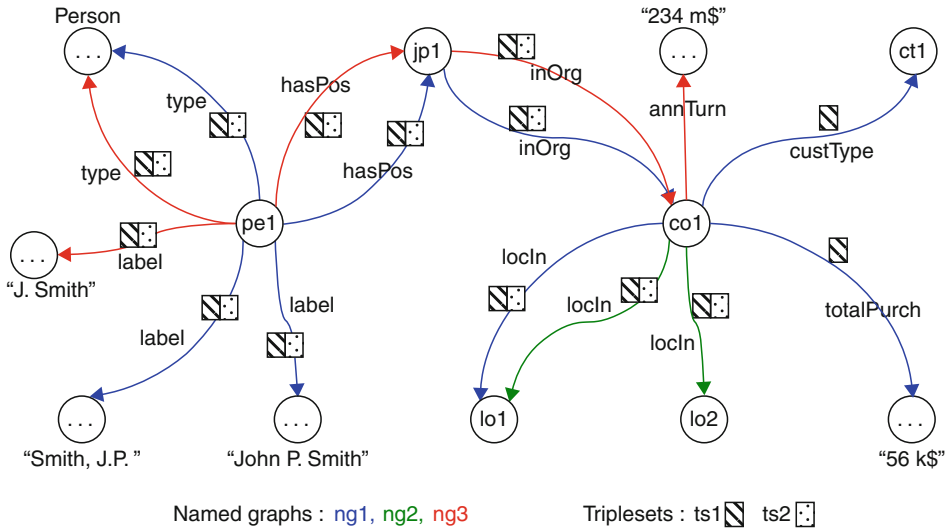
$$\{ G, (\langle U1 \rangle, G1), (\langle U2 \rangle, G2), \dots (\langle Un \rangle, Gn) \}$$

where G and each G_i are RDF graphs, and each $\langle U_i \rangle$ is a distinct IRI (internationalized URI). The pairs $(\langle U_i \rangle, G_i)$ are called *named graphs*, where $\langle U_i \rangle$ is the name of graph G_i . G is called *default graph* – it contains all triples, which belong to the dataset, but not to any specific named graph. The notion of default graph is not present in [41]. Intuitively, a *dataset* integrates several RDF graphs in such a way that each graph can be distinguished, manipulated, and addressed separately. In a nutshell, in the RDF data model, extended with named graphs, statements can be part of named graphs, which can be used to model provenance or other contextual meta-information: Named graphs are also referred to as *contexts* in some systems.

Formally, a dataset can be represented as an RDF multi-graph, which, in its turn, can be represented as a set of quadruples of the following type: $\langle S, P, O, G \rangle$. The first three elements of the quadruple, $\langle S, P, O \rangle$, represent an RDF statement; the fourth element, G , represents the name of the named graph.

The SPARQL specification does not provide sufficient formal grounds for the semantics of the named graphs in order to determine the possible behavior of a semantic repository that supports such an extended RDF model. As SPARQL supports no data modification, it is unclear what should be the formal consequences of adding or removing a statement from a named graph. Counting statements in a SPARQL dataset is also not specified. To fill this gap, the specification of the second generation of the ORDI framework, [43] defined these aspects of the semantics of named graphs and introduced a new notion, namely triplesets.

A *tripleaset*, as introduced in [43], is a mechanism to deal with parts of datasets or to group some of the statements in a dataset. An RDF dataset with named graphs and



■ Fig. 7.4

RDF graph with named graphs and triplets

triple sets is depicted in Fig. 7.4. The difference between named graphs and datasets can be explained as follows:

- Named graphs “own” the statements; for example, each statement belongs to a specific named graph of the default graph. When a statement is added or deleted from a named graph, a new arc appears or disappears from the multi-graph, which represents the datasets; the count of the arcs increases or decreases, respectively.
- Triplets are tags on the statements. When a statement is associated with a triplet, this can be seen as an association operation, which does not add a new arc in the graph. When statement is removed from a triplet, that is, it is no longer a member of this group of statements, it does not disappear from the dataset, it is just being un-tagged or disassociated.

Given the above extensions, the atomic entity of the triplet model is a quintuple:

$$\langle S, P, O, G, \{TS_1, \dots, TS_n\} \rangle$$

where G is the named graph and $\{TS_1, \dots, TS_n\}$ is a set of identifiers of the triplets to which the contextualized statement $\langle S, P, O, G \rangle$ is associated. In other words, each statement (from each graph) can be member of multiple triplets. Formally, the extension of a triplet is an RDF multi-graph, a subset of the set of all quadruples in the dataset. Triplets are named, that is, each triplet is associated with an URI. It is worth noting that the above quintuple is provided for the sake of formal specification of the semantics of the extended RDF data model. Semantic repositories can (and most of them do) implement alternative data representation and indexing structures, while supporting the same semantics.

The need for the enhancement of the RDF data model with triplesets is a result from the clear specification of the semantics of the named graphs. Named graphs are used most often for tracking of provenance, for example, when multiple graphs from different sources are merged or referenced (e.g., when dealing with linked data, see [Sects. 7.1.4](#) and [7.8](#)). In such a scenario, strong “ownership” semantics should be enforced for the named graphs so that updates of the contents of specific named graphs can have real impact on the contents of the dataset. Once named graphs are given such semantics, there is a need for a mechanism which allows for dealing with metadata about the contents of an integrated dataset. Triplesets are defined as a weaker mechanism to group quadruples (statements form a dataset) and assign metadata with them. Moreover, since the triplesets allow for the designation or tagging of parts of a dataset, they are especially useful when selecting parts of the dataset, for example, in the course of multistage processing, where intermediate results should be passed from one component to another. Triplesets are supported by BigOWLIM and the storage infrastructure of Freebase (see [Sect. 7.8.1.1](#)); they are also a standard feature of the LarkC data layer (see [Sect. 7.5.1](#) in [31]). A more concise description of the tripleset mechanism is provided in [35].

7.1.4 Linked Data

The notion of “linked data” (linked data principles and applications are presented in greater detail in [Semantic Annotation and Retrieval: Web of Data](#); this section provides only a brief introduction to their major principles as linked data bring specific requirements for semantic repositories) is defined by Tim Berners-Lee, [4], as RDF graphs, published on the WWW so that one can explore them across servers by following the links in the graph in a manner similar to the way the HTML Web is navigated. It is viewed as a method for exposing, sharing, and connecting pieces of data, information, and knowledge on the Semantic Web using URIs and RDF. “Linked data” are constituted by publishing and interlinking open data sources, following four principles. These are:

1. Using URIs as names for things
2. Using HTTP URIs, so that people can look up those names
3. Providing useful information when someone looks up a URI
4. Including links to other URI, so people can discover more things

In fact, most of the RDF datasets fulfill principles 1, 2, and 4 by design. The piece of novelty in the design principles above concerns the requirement for enabling Semantic Web browsers to load HTTP descriptions of RDF resources based on their URIs. To this end, data publishers should make sure that:

- The “physical” addresses of the published pieces of data are the same as the “logical” addresses, used as RDF identifiers (URIs).
- Upon receiving an HTTP request, the server should return an RDF-molecule, that is, the set of triples that describe the resource.

Linking Open Data (LOD, [68]) is a W3C SWEO community project aiming to extend the Web by publishing open datasets as RDF and by creating RDF links between data items from different data sources. Linked Open Data provides sets of referencable, semantically interlinked resources with defined meaning. The central dataset of the LOD is DBPedia – an RDF extract of the Wikipedia open encyclopedia (DBPedia is discussed in [▶ Sect. 7.5.3](#)). Because of the many mappings between other LOD datasets and DBPedia, the latter serves as a sort of a hub in the LOD graph assuring a certain level of connectivity. LOD is rapidly growing – as of September 2010, it contains 203 datasets, with total volume of 25 billion statements, interlinked with 142 million statements as illustrated on [▶ Fig. 7.5](#).

Although not related to semantics, the linked data concept turns into an enabling factor for the realization of the Semantic Web as a global Web of structured data around the Linking Open Data initiative. Still, querying and reasoning with linked data raises various challenges related to the very scale and nature of such data. A specific approach for the management of linked data, named “reason-able views,” is presented in [▶ Sect. 7.8.1](#) and [▶ Sect. 7.8.1.1](#) provides an overview of few of the central LOD datasets.

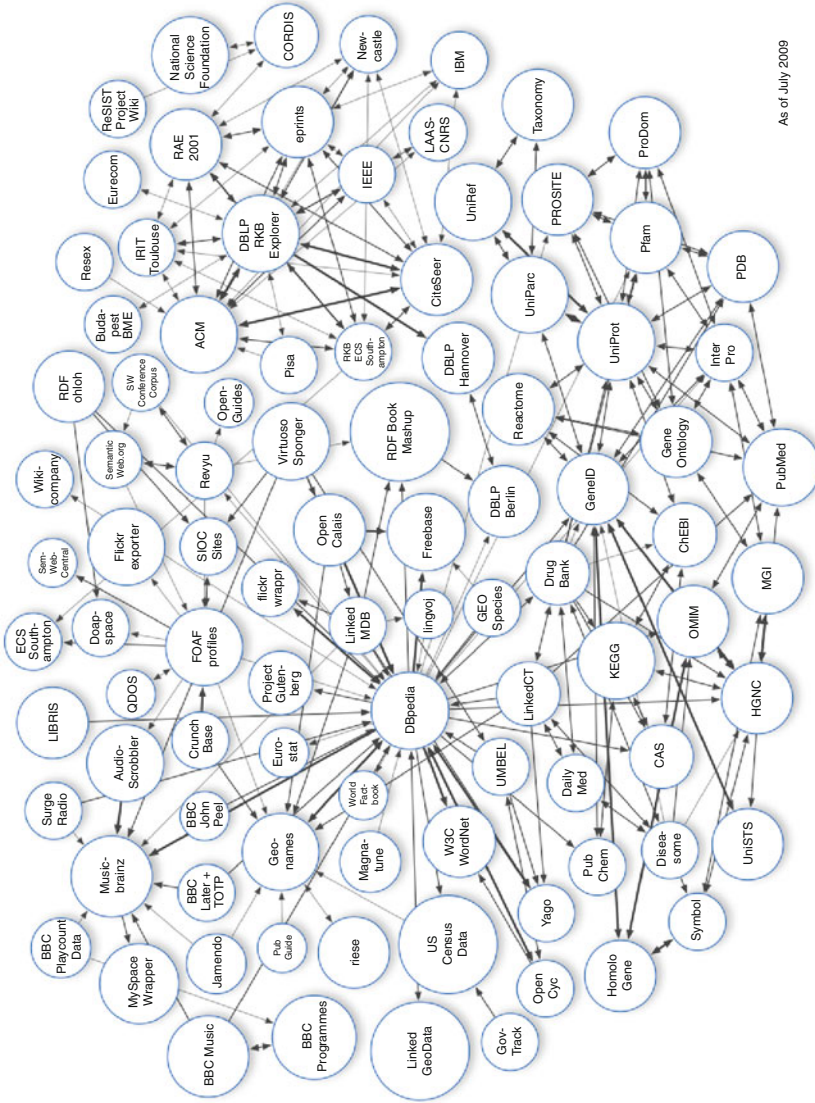
More generally, the management and publishing of linked data creates major usage scenarios for semantic repositories, which bring a range of specific requirements, such as:

- Dealing with a massive number of different predicates with no proper definition – there are about hundred thousand predicates in DBPedia.
- Optimizations in the reasoning with owl:sameAs-equivalence (see [▶ Sect. 7.2.1.4](#)) – in linked data, many objects have multiple different identifiers across different datasets or even within a single dataset.
- Novel query methods need to be developed as the standard structured query languages and engines assume schema knowledge at the time of query specification. In the linked data scenario, such assumptions are nonrealistic; methods of the type of the RDF Search developed in BigOWLIM and used in FactForge are required (see [▶ Sects. 7.6.4](#) and [▶ 7.8.1.1](#) respectively).

7.2 Reasoning in the Semantic Repositories

A major distinctive feature of the semantic repositories, versus most of the other database management systems (DBMS), is that they manage the data taking into consideration their semantics. They can interpret the semantics of the schemas and the data loaded into them and deliver answers, based on this interpretation. In the simplest scenario, a request using the pattern “?x fellowCitizenOf Frank” will return Orri as a result, if “Frank fellowCitizenOf Orri” was asserted and fellowCitizenOf is defined to be a symmetric relationship.

The ability of the semantic repositories to interpret the semantics of the data delivers one major advantage. The syntax of the query is no longer required to match the syntax of



As of July 2009

Fig. 7.5 Map of the datasets in Linking Open Data (LOD) project, [68]

the assertion. In the example above, the user will receive one and the same results, disregarding the direction in which the symmetric relationship was asserted. A slightly more complicated scenario is presented in [Fig. 7.1](#), where the request is formulated through a more general relationship (relative), but the repository is capable of delivering results based on assertion of a more specific one (child). More generally, the engine takes care to interpret the semantics of the query and the semantics of the data in order to deliver all correct results.

The ability to abstract the query syntax from the data syntax bears important advantages in data access scenarios where one has to deal with complex relationships or with schema diversity. [Figure 7.1](#) presented a simple example of enhanced analytical power – one can obtain results at the desired level of generality, even if the underlying data are far more specific. As long as the semantic repositories can interpret the semantics in a recursive fashion, one can enjoy interpretations of the data, which combine results from previous interpretations and explicit assertions. In other words, depending on the data patterns and the semantics, one can retrieve facts, which are the results of multiple steps of interpretation, and by this way one can uncover relationships, which would otherwise remain hidden.

In a data integration scenario, often similar facts are encoded in different ways across different data sources. Imagine a case when one data source deals with information about the locations of airports encoded as “Stansted airportOf London”, while another data source associates airports to cities in a more fine grained manner, using patterns like “London hasMainAirport Heathrow” and “London hasAlternativeAirport Gatwick.” An approach to integrate these data using RDFS and OWL would be to define a new property `hasAirport`, and to define it to be inverse property of `airportOf` and super-property of both `hasMainAirport` and `hasAlternativeAirport`. Given such semantic vocabulary alignment, a semantic repository will be able to return all the three airports as a match for the retrieval pattern “London hasAirport ?x.” This way data and schema interpretation facilitate data integration.

Although all the examples of data or query interpretation can be achieved when working with other types of DBMS also, RDF-based semantic repositories provide the most efficient and standard compliant mechanism to deliver such limited intelligence in a robust, reliable, and manageable manner.

7.2.1 Lightweight Inference Integration

Semantic repositories are expected to demonstrate performance and scalability comparable to those of the mature database management systems. They should be able to deal with billions of facts, to handle online updates while at the same time processing vast query loads. This puts heavy constraints on both the worst-case and average-case complexity of the reasoning algorithms to be used. Further, such repositories shall be used and operated by engineers and administrators with no deep understanding of reasoning or

mathematical logic. This means that the inference techniques used should allow inferences to be traced and “debugged” by a typical database administrator, given one week of RDF (S) and OWL training. Altogether, these factors limit the applicability of non-tractable languages and techniques like satisfiability checking in semantic repositories. As seen in [▶ Sect. 7.6](#), most of the semantic repositories (which offer inference at all) support reasoning based on Datalog-style rule entailment.

7.2.1.1 Rule-Based Entailment Example and Relations to Query Evaluation

The following example is meant to facilitate intuitive understanding of the mechanisms for Datalog-style rule entailment in RDF repositories (a proper definition of RDF-based rule-entailment formalisms is provided in [▶ Sect. 7.1.1](#)) and their similarities to the structured query evaluation mechanisms. In a typical language of this type, rules are defined through premises and consequences. One or more RDF triple patterns, involving variables, constitute the premises of the rule and several others represent the consequences. For variable bindings, which satisfy the premises, the reasoner can infer the consequences. For instance, given the rule:

```
<I, rdf:type, C1 > AND <C1, rdfs:subClassOf, C2> => <I, rdf:type, C2>
```

where *I*, *C1*, and *C2* are variables, if the following statements are already in the repository

```
<myData:Maria, rdf:type, ptop:Woman>
<ptop:Woman, rdfs:subClassOf, ptop:Person>
```

it can infer the following statement

```
<myData:Maria, rdf:type, ptop:Person>
```

It is important to be considered that rule-based reasoning of the type presented above is computationally very similar to the evaluation of structured queries in languages like SQL and SPARQL (see [▶ Sect. 7.1.3](#)). For instance, the consequences of the above rule are the same as the results of the following query (The definitions of the namespace prefixes *rdf* and *rdfs* are omitted above for the sake of better readability.):

```
CONSTRUCT { ?I rdf:type ?C2 }
WHERE { ?I rdf:type ?C1. ?C1 rdfs:subClassOf ?C2 }
```

The latter has several implications. As a start, it indicates that reasoning can be implemented by means of query evaluation. It is also the case that most of the performance considerations concerning query evaluation are also relevant to rule-based reasoning – such is the case with distribution approaches discussed in [▶ Sect. 7.4.2](#).

7.2.1.2 Reasoning Strategies: Forward- and Backward-Chaining

The main strategies for the implementation of rule-based inference are as follows. (More information on rule-based semantics can be found in [▶ KR and Reasoning on the Semantic Web: RIF](#), which introduces the Rule Interchange Format and provides discussion on interoperability across multiple rule-based systems.)

- **Forward-chaining:** to start from the known facts (the explicit statements) and to derive valid implications. The goals of such reasoning can vary: to compute the *inferred closure*; to answer a particular query; to infer a particular sort of knowledge (e.g., the class taxonomy).
- **Backward-chaining:** to start from a particular fact or a query and to verify it or to find all possible solutions. In a nutshell, the reasoner decomposes or transforms the query into simpler (or alternative) requests that can be matched directly to explicit facts available in the KB or can be proven through further recursive transformations. The most popular example for backward-chaining is the unification mechanism in Prolog.

In the context of forward-chaining, *inferred closure* (also known as *deductive closure*) is the extension of a knowledge base (or an RDF graph) with all the implicit facts (or statements) that could be inferred from it, based on the enforced semantics.

In relational database management systems, *materialized views* represent an approach where the results of the query, which define the view, are cached in a sort of temporary table. Generally, in inference engines, *materialization* is the process of making some reasoning results explicit. In both cases, it is a matter of storing the results of data processing so that these are available at a later stage. In relation to semantic repositories, materialization is used to refer to a range of techniques similar to those referred above, which are, however, diverse and not standardized. Below is provided a definition for materialization, as a reasoning strategy for semantic repositories, considering that such a strategy should be transparent for the clients of the repository. In other words, following the “separation of concerns” principle, the clients of the repository should not experience functional differences in the case of a change of the strategy. Given the same sequence of update transactions and queries, the repository should return the same results for the queries disregarding the reasoning strategy, which is implemented. Still the clients can (and are likely to) experience change in some nonfunctional parameters, for example, the speed of specific operations.

Materialization is a strategy for reasoning in semantic repositories where the inferred closure of the contents of the repository is derived through forward-chaining and maintained in a form that allows its efficient usage. In practice, this means that the inferred closure of large volumes of data should be persisted and indexed in order to enable efficient query evaluation or retrieval. The repository should take care to keep the inferred closure up-to-date after each update of its contents, that is, at the end of each update transaction.

7.2.1.3 The Advantages and Applicability of the Different Strategies

The principal advantages and disadvantages of materialization can be summarized as follows:

- The loading of new facts gets slower because the repository is extending the inferred closure after each transaction. In fact, all the reasoning is performed during loading.
- The deletion of facts is also slow, because the repository should remove from the inferred closure all the facts, which cannot be inferred any longer.
- The maintenance of the inferred closure usually requires considerable additional space (RAM, disk, or both, depending on the implementation).
- Query and retrieval are fast, because no deduction, satisfiability checking, or other sorts of reasoning are required.

The advantages and disadvantages of the backward-chaining-based interpretation of the data in semantic repositories are also well known:

- The loading and modification of the data are faster, compared to repositories using materialization, because no time and space is lost for the computation and maintenance of the inferred closure of the data.
- Query evaluation is slower because extensive query rewriting (reformulation and expansion) has to be performed. As a result, a potentially much larger number of lookups in the indices are required, as compared to standard query evaluation.

The choice of the most appropriate and efficient reasoning strategy requires balancing between loading and query performance, considering several factors, related to the data, its semantics, and the typical usage scenarios and loads:

- When the data are updated very intensively, there will be relatively high costs for the maintenance of the inferred closure, so materialization becomes less attractive.
- In the case of challenging query loads, backward-chaining becomes inappropriate, as long as it increases the time for query evaluation considerably.
- If materialization requires time and space, which are hard to be secured, it should be avoided.
- Whenever low response time has to be guaranteed, backward-chaining should be avoided because it leads to recursive query evaluation, where performance cannot be easily estimated and managed.

The selection of the appropriate reasoning strategy for the specific application is very important as long as it can change the performance of some operations several times; see for instance, the variation in the loading performance in dependence of the materialization complexity in [Sect. 7.7.2](#). One should note, however, that the reasoning strategy generally impacts the so-called average-case complexity. The worst-case reasoning complexity, even for relatively simple ontology languages, is unbearable for semantic

repositories, independently of the selected strategy. Suppose, for instance, a dataset where each of the one million citizens of Amsterdam is linked to one of the other citizens with a `fellowCitizen` relationship. Suppose also that the latter is defined to be transitive and symmetric, as it naturally is. The inferred closure of such dataset will contain one trillion facts; thus, total materialization is impractical. A query checking the pattern “`AmsCitizen fellowCitizen ?x,`” in a system using backward-chaining, will have to make one million steps of recursion, which is also practically unfeasible. Avoiding such cases requires through ontology and schema design and, more generally, concise data modeling. Architects and database administrators should be aware of such dangers and take care to control the complexity. In this respect, semantic repositories are not very different from the mainstream DBMS, which also expose unbearable query evaluation performance in cases of poor data or query modeling – no one can guarantee the good performance of an SQL database if indices are not properly defined or if a query specifies unconstrained Cartesian products on large tables.

Total materialization is adopted as a reasoning strategy in a number of popular Semantic Web repositories, including Sesame, Jena, DAML DB, ORACLE, and BigOWLIM (please, refer to the corresponding subsections of [Sect. 7.6](#) for details). Probably, the most important advantage of the usage of materialization in semantic repositories is that all data are available at query time, which makes possible RDBMS-like query optimization techniques. The query evaluation engine can use statistics to make guesses about the cost of evaluation of a particular constraint and the cardinality of its results; such guesses allow DBMS to reorder constraints in order to build an efficient query evaluation plan. Such optimization techniques are far more complex in the case of deductive query evaluation.

7.2.1.4 Hybrid Strategies and Dynamic Materialization

A range of different techniques have been invented to improve the efficiency of the reasoning strategies for specific types of data and usage scenarios.

There are cases where hybrid strategies, which involve *partial materialization* and partial backward-chaining for specific data patterns, deliver the best overall performance. One such schema, related to the efficient handling of the semantics of `owl:sameAs`, is described in [Sect. 7.4.1](#). The lightweight version of OWLIM (SwiftOWLIM ver. 2.9, see [Sect. 7.6.4](#)) implements a partial materialization schema where the inference related to transitive, inverse, and symmetric properties is performed during forward-chaining; however, a specific materialization strategy is applied to avoid the inflation of the indices of the repository.

There are repositories, which use backward-chaining, but do caching of the intermediate results, which effectively means that they perform partial materialization on demand. Such a strategy, named *dynamic materialization*, is implemented in AllegroGraph (see [Sect. 7.6.2](#)). The advantage, compared to the full materialization, is that the inferred closure is materialized incrementally. Further, in many application

scenarios, a significant fraction of the implicitly inferable facts will never be accessed during the handling of retrieval queries – in the case of dynamic materialization, those will never be materialized. The advantage compared to backward-chaining is that the same conclusions do not need to be inferred multiple times. The principal disadvantages of the dynamic materialization are that (1) it still slows down the query processing, (2) query optimization, based on cardinality statistics, is still hardly applicable, and (3) the cached partial materialization still requires maintenance upon modification of the repository contents.

7.2.1.5 The Honey and the Sting of owl:sameAs

owl:sameAs is a system predicate in OWL, declaring that two different URIs denote one and the same resource. Most often it is used to align the different identifiers of the same real-world entity used in different data sources. In FactForge (see ▶ Sect. 7.8.1.1), one and the same entity has different URIs in the different linked data LOD datasets (See ▶ Sect. 7.1.4) where it appears. For instance, in DBpedia, the URI of the city of Vienna is <http://dbpedia.org/page/Vienna>, while in Geonames, it is <http://sws.geonames.org/2761369/>. DBpedia contains the statement

```
(S1) dbpedia:Vienna owl:sameAs geonames:2761369
```

which declares that the two URIs are equivalent. owl:sameAs is probably the most important OWL predicate when it comes to merging data from different data sources.

Following the specification of OWL (To be more specific, this is the semantics defined in OWL 2 RL, but also in the other dialects discussed in ▶ Sect. 7.1.1.), whenever two URIs U_1 and U_2 are declared equivalent, all statements that involve U_1 and are true will also be inferable and retrievable, with U_2 , replacing U_1 at the same position. For instance, in Geonames, the city of Vienna is defined as part of <http://www.geonames.org/2761367/> (the first-order administrative division in Austria with the same name), which, in turn, is part of Austria (<http://www.geonames.org/2782113/>):

```
(S2) geonames:2761369 gno:parentFeature geonames:2761367
```

```
(S3) geonames:2761367 gno:parentFeature geonames:2782113
```

As long as gno:parentFeature is a transitive relationship, in the course of the initial forward-chaining, it will be derived that the city of Vienna is also part of Austria:

```
(S4) geonames:2761369 gno:parentFeature geonames:2782113
```

Based on the semantics of owl:sameAs, from (S1), it should be inferred that statements (S2) and (S4) also hold for Vienna when it is referred with its DBpedia URI:

```
(S5) dbpedia:Vienna gno:parentFeature geonames:2761367
```

```
(S6) dbpedia:Vienna gno:parentFeature geonames:2782113
```

These are true statements and when querying RDF data, no matter which one of the equivalent URIs is used in the explicit statements, the same results will be returned. When one considers that Austria, too, has an equivalent URI in DBpedia.

```
(S7) geonames:2782113 owl:sameAs dbpedia:Austria
```

it should also infer that:

```
(S8) dbpedia:Vienna gno:parentFeature dbpedia:Austria
```

```
(S9) geonames:2761369 gno:parentFeature dbpedia:Austria
```

```
(S10) geonames:2761367 gno:parentFeature dbpedia:Austria
```

In the above example, there are two alignment statements (S1 and S7), two statements carrying specific factual knowledge (S2 and S3), one statement inferred due to a transitive property (S4), and seven statements inferred as a result of `owl:sameAs` alignment (S5, S7, S8, S9, S10, and the inverse statements of S1 and S7, which are not given above due to space limitations). As seen, inference without `owl:sameAs` inflated the dataset by 25% (one new statement on top of four explicit), while `owl:sameAs` related inference increased the dataset by 175% (seven new statements). Considering that Vienna has a URI also in UMBEL, which is also declared equivalent to the one in DBpedia, the addition of one more explicit statement for this alignment will cause inference of four new implicit statements (duplicates of S1, S5, S6, and S8). Although this is a small example, it provides a good indication about the performance implications of using `owl:sameAs` alignment in LOD. Also, because `owl:sameAs` is a transitive, reflexive, and symmetric relationship, a set of N equivalent URIs N^2 `owl:sameAs` statements will be generated for each pair of URIs. Thus, although `owl:sameAs` is useful for interlinking RDF datasets, its semantics cause considerable inflation of the number of implicit facts that should be considered during inference and query evaluation (either through forward- or backward-chaining).

To overcome this problem, BigOWLIM and ORACLE (see [▶ Sects. 7.6.4](#) and [▶ 7.6.7](#)) handle `owl:sameAs` in a specific manner. In their indices, each set of equivalent URIs (the equivalence class with respect to `owl:sameAs`) is represented by a single super-node. This way, the repository does not inflate the indices and, at the same time, retains the ability to enumerate all statements that should be inferred using the equivalence upon retrieval request (e.g., during inference or query evaluation).

7.2.1.6 Truth Maintenance and Smooth Invalidation

Normally, repositories, which use materialization support dialects of OWL, which allow for monotonic inference (see [▶ Sect. 7.2.2](#)). Upon the addition of new statements, they can incrementally extend the inferred closure with new facts, which can be entailed from the new data. The deletion of statements requires the repository to remove from the inferred closure statements, which are no longer inferable. The classical approach to

implement this is known from the expert systems as a *truth maintenance system* (TMS), which keeps meta-information about which fact can be inferred from which other facts. While such systems allow for the tracing of the inference dependencies and the invalidation of the statements which are no longer supported, the overheads associated with the maintenance of the TMS information itself appear quite high for most of the scenarios of usage of semantic repositories. This is the reason why most of the repositories which use materialization do not implement a TMS – upon deletion, they have to delete the entire inferred closure and compute it again. Scenarios where deletion can be avoided or postponed have no problem with such strategy. However, there are plenty of scenarios where deletion should be performed in real time.

Smooth invalidation is technique implemented in BigOWLIM (see [Sect. 7.6.4](#)), which allows for the efficient maintenance of the inferred closure upon deletion. It is based on an algorithm which performs a sequence of backward- and forward-chaining iterations to figure out what part of the deductive closure is no longer supported, without using truth maintenance information. The complexity of this algorithm is comparable to the complexity of the plain forward-chaining. In other words, the update of the inferred closure after removal of some statements requires time and computational resource comparable to those needed for the update of the inferred closure after the addition of the same statements.

One should note that the removal of key statements from the schema may lead to the invalidation of a large fraction of the inferred closure – in such cases the computation of the inferred closure from scratch can be the more efficient option. Still, in most of the data management scenarios, non-monotonic schema changes represent a tiny fraction of all update transactions and usually can be postponed and implemented in a time slot when full re-inference is feasible. This way, for a very large range of applications, forward-chaining, combined with optimizations (like those discussed in [Sects. 7.2.1.3](#) and [7.2.1.4](#)) and smooth invalidation, represents the optimal reasoning strategy, which delivers good performance through the entire life cycle of the data. A BigOWLIM instance implementing this strategy was used behind the BBC's World Cup 2010 website, presented in [Sect. 7.8.2](#).

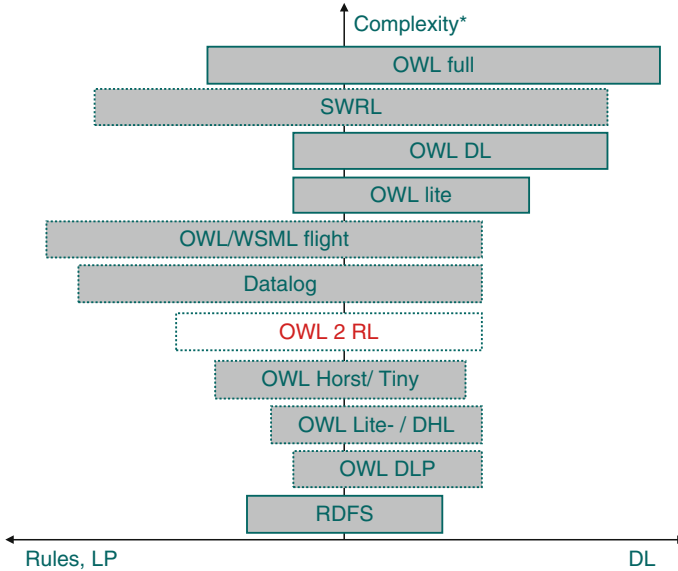
7.2.2 OWL Dialects Suitable for Scalable Inference

In order to match the expectations for the next-generation global Web of data, the Semantic Web requires a scalable high-performance storage and reasoning infrastructure. One challenge toward building such an infrastructure is the expressivity of its schema and ontology definition standards RDFS and OWL. RDFS, [10], is the schema language for RDF, which allows for the definitions of subsumption hierarchies of classes and properties; the latter being binary relationships defined with their domains and ranges. While RDFS is generally a fairly simple knowledge representation language, implementing semantic repositories which support its semantics and provide performance and scalability comparable to those of relational database management systems (RDBMS) is very challenging.

The semantics of RDFS is based on Logical Programming (LP) – a declarative programming paradigm, in which the program specifies a computation by giving the properties of a correct answer. The LP languages like PROLOG emphasize the logical properties of a computation, using logic and proof procedures to define and resolve problems. Most logic programming is based on Horn-clause logic with negation-as-failure to store the information and rule entailment to solve problems. Datalog is a query and rule language, a simplified version of PROLOG, meant to enable the efficient implementation of deductive databases. The semantics of RDFS is defined by means of rule-entailment formalism, which is a simplification of Datalog.

OWL [13] is an ontology language which supports more comprehensive logical descriptions of the schema elements, for instance: transitive, symmetric, and inverse properties; unions and intersections of classes; and property restrictions (for a detailed introduction into OWL one can refer to [▶ KR and Reasoning on the Semantic Web: OWL](#)). The first version of the OWL specification, which was published as a W3C standard in year 2004 has three dialects: OWL Lite, OWL DL, and OWL Full. They range in their levels of expressivity. OWL Lite is a subset of OWL DL, and OWL DL is a subset of OWL Full. The OWL language is based on description logics (DL, [3]). The reasoning procedures of DLs are decision procedures that are aimed to always terminate – in mathematical logic terms this means that DLs are decidable. Compared to other logical languages, DLs are relatively inexpressive. Still reasoning with DLs is based on satisfiability checking, which means that in order to prove or to reject a specific statement, a DL reasoner needs to check whether it is possible or not to build a model of the world that satisfies a “theory” which includes this statement or its negation. For instance, suppose that there is a semantic repository which contains one billion statements and a client makes a query, checking whether a specific resource is an instance of a specific class. In order to validate this, with respect to the semantics of OWL DL, a repository should add to its current contents the statement that the resource is not an instance of the class and check whether the new state of the repository is consistent. It is clear that such semantics is impractical to implement for large volumes of data. Even the simplest dialect of OWL, OWL Lite is a DL formalism which does not support algorithms enabling efficient inference and query answering over reasonably large knowledge bases.

Logic programming and description logics support semantics and data interpretation capabilities of a different nature: LP uses rules to infer new knowledge, whereas DL employ descriptive classification mechanisms. None of these is more powerful or expressive than the other one – there are meaning aspects, which can be expressed in each one of them, which cannot be expressed in a language from the other paradigm. As a result, the semantics of OWL Lite and DL are incompatible with that of RDFS. (The issues related to the interoperability and layering of the Semantic Web languages are also discussed in [▶ Introduction to the Semantic Web Technologies](#).) Although OWL was meant to be layered on top of RDFS in the Semantic Web specification stack, there is no “backward compatibility.” In practical terms, this means that it may be impossible to “upgrade” to OWL an application, which uses RDFS schemas, without replacing them with OWL



■ Fig. 7.6

Diagram of expressivity of OWL dialects

ontologies. The latter may require considerable changes in the semantics of the classes and the properties and in the data modeling principles used in the application.

To bridge the gap of expressivity, compatibility, and logical decidability and reach the goals of scalable inference, other dialects of OWL have been created which lie between RDF(S) and OWL Lite. Figure 7.6 presents a simplified map of the expressivity or complexity of a number of these OWL-related languages together with their bias toward description logic (DL) and Logical Programming (LP) based semantics. The diagram provides a very rough idea about the expressivity of the languages, based on the complexity of entailment algorithms for them. A direct comparison between the different languages is impossible in many of the cases. For instance, Datalog is not simpler than OWL DL, it just allows for a different type of complexity.

OWL DLP is a nonstandard dialect, offering a promising compromise between expressive power, efficient reasoning, and compatibility. It is defined in [21] as the intersection of the expressivity of OWL DL and Logic Programming (LP). In fact, OWL DLP is defined as the most expressive sub-language of OWL DL, which can be mapped to Datalog. OWL DLP is simpler than OWL Lite. The alignment of its semantics to the one of RDFS is easier, as compared to the Lite and DL dialects. Still, this can only be achieved through the enforcement of some additional modeling constraints and transformations. A broad collection of information related to OWL DLP can be found in [49]. DLP has certain advantages:

- There is freedom to use either DL or LP (and associated tools and methodologies) for modeling purposes, depending on the modeler's experience and preferences.

- From an implementation perspective, either DL reasoners or deductive rule systems can be used. Thus it is possible to model using one paradigm, for example, a DL-biased ontology editor, and to use a reasoning engine based on the other paradigm, for example, a semantic repository based on rules.

These features of DLP provide extra flexibility and ensure interoperability with a variety of tools. Experience with using OWL has shown that existing ontologies frequently use only very few constructs outside the DLP language.

In [62] ter Horst defines RDFS extensions toward rule support and describes a fragment of OWL, more expressive than OWL DLP. He introduces the notion of R-entailment of one (target) RDF graph from another (source) RDF graph on the basis of a set of entailment rules R . R-entailment is more general than the D-entailment used by Hayes, [26], in defining the standard RDFS semantics. Each rule has a set of premises, which conjunctively define the body of the rule. The premises are “extended” RDF statements, where variables can take any of the three positions. The head of the rule comprises one or more consequences, each of which is, again, an extended RDF statement. The consequences may not contain free variables, that is, which are not used in the body of the rule. The consequences may contain blank nodes.

The extension of the R-entailment (as compared to the D-entailment) is that it “operates” on top of the so-called generalized RDF graphs, where blank nodes can appear as predicates. R-entailment rules without premises are used to declare axiomatic statements. Rules without consequences are used to imply inconsistency.

This extension of RDFS became popular as “OWL Horst.” As outlined in [62], this language has a number of important characteristics:

- It is a proper (backward-compatible) extension of RDFS. In contrast to OWL DLP, it puts no constraints on the RDFS semantics. The widely discussed meta-classes (classes as instances of other classes) are not disallowed in OWL Horst.
- Unlike the DL-based rule languages, like SWRL, [28] and [46], R-entailment provides a formalism for rule extensions without DL-related constraints;
- Its complexity is lower than the one of SWRL and other approaches combining DL ontologies with rules; see [♦ Sect. 7.5](#) of [62].

OWL Horst is supported by OWLIM and ORACLE (presented in [♦ Sect. 7.6](#)), which makes it the OWL dialect that has the largest industry support. An official OWL dialect with the same properties emerged recently under the name OWL 2 RL. The latter is one of the tractable profiles (dialects) defined in the specification of OWL 2, [45] – the next version of the OWL language that is currently in process of standardization. OWL 2 RL is designed with the objective to be the most expressive OWL dialect, which allows for efficient reasoning with large volumes of data in rule-based systems. OWL 2 RL was inspired by OWL Horst – its semantics is defined with the rule language equivalent to R-entailment. However, OWL 2 RL is considerably more expressive than OWL Horst. Support for OWL 2 RL is provided by several reasoning engines, including OWLIM and ORACLE. More details on the OWL 2 profiles can be found in [♦ KR and Reasoning on the Semantic Web: OWL](#).

Recent research reported in [58] evaluates the level of completeness of the inference supported by few inference engines (namely, HAWK) and semantic repositories: IBM's Minerva, Sesame (▶ Sect. 7.6.8), and OWLIM (▶ Sect. 7.6.4). It demonstrates that although OWLIM supports sufficient reasoning to answer the LUBM (LUBM benchmark is introduced in ▶ Sect. 7.5.1) queries correctly, it is still not complete with respect to the semantics of the data and the queries.

7.3 Semantic Repository Tasks, Performance Factors, and Dimensions

Measuring and benchmarking the performance of semantic repositories is an important aspect in allowing the engineers to understand and use them efficiently. As discussed in ▶ Sect. 7.1.2, semantic repositories are RDF databases that may or may not provide lightweight inference support. Their benchmarking is a complicated exercise, which requires a clear conceptualization and structuring. This section provides a conceptual framework for benchmarking of semantic repositories, as further development of [32]. The framework takes into consideration the tasks executed by the semantic repository, examines the performance factors and the performance dimensions of the measurements, and introduces the concept of full-cycle benchmarking.

A wide range of links to resources related to benchmarking RDF repositories can be found on the page on “RDF Store Benchmarking” in the ESW wiki, maintained by W3C, at <http://esw.w3.org/topic/RdfStoreBenchmarking>.

7.3.1 Tasks

The major tasks and activities toward which the performance of semantic repositories needs to be benchmarked are:

- Data loading, including parsing, persistence, and indexing of both instance data and ontologies
- Query evaluation, including query preparation, optimization, and fetching
- Data modification, which may involve changes to the ontologies and the schemas

Inference is not a first-level activity in a semantic repository. Depending on the implementation, it can affect the performance of the other activities, for instance, when inference is performed during loading.

Modifications to the data and/or schemas (e.g., updating and deleting values or changing class definitions) represent another important class of the tasks performed against a semantic repository. Most of the contemporary RDF-related benchmark suites do not cover modification tasks, because their specifics, complexity, and importance can change considerably across applications and usage patterns.

7.3.2 Performance Factors

The performance of data loading depends on several factors:

- *Materialization* – whether and to what extent forward-chaining is performed at load time, including the complexity of the forward-chaining (see [Sect. 7.2.1](#))
- *Data model complexity* – support for extended RDF data models (see [Sect. 7.1.3](#)), for instance, including named graphs, is computationally more “expensive” as compared to the simple triple model
- *Indexing specifics* – repositories may or may not create a variety of different indices in dependence of the datasets loaded, the foreseen usage patterns, hardware constraints, etc.
- *Data access and location* – where the data is imported from, for instance, local files, loaded from the network, etc.

Several factors affect the time and memory space, or more generally, the computing resources, required for query evaluation:

- *Deduction* – whether and to what extent backward-chaining is involved, whether it is recursive, etc. (see [Sect. 7.2.1](#))
- *Size of the result-set* – fetching large result-sets can take considerable time
- *Query complexity* – the number of constraints (e.g., triple-pattern joins), the semantics of the query (e.g., negation- and disjunction-related clauses), the usage of operators that are hard to support through indexing (e.g., LIKE)
- *Number of clients* – number of simultaneous client requests
- *Quality of results* – what is the quality of the results required in modalities where incomplete answers are requested

Transaction size and level of isolation may also have serious impact on the performance of both loading and query evaluation. Although many semantic repositories provide some sort of transaction isolation, it is usually less comprehensive than the corresponding mechanisms in the mature RDBMS. Furthermore, transaction management in large-scale systems is usually carefully designed and tuned for each specific setup.

7.3.3 Performance Dimensions

The performance dimensions of a semantic repository comprise parameters of a specific task or scenario, which affect its speed, and the size of the loaded datasets. There are several parameters affecting the speed of a semantic repository:

- *Scale* – the size of the repository in terms of number of RDF triples (more generally, facts or atomic assertions).

- *Schema and data complexity* – the complexity of the ontology/logical language, the specific ontology (or schema), and the dataset. A highly interconnected dataset, with long chains of transitive properties, can appear far more challenging for reasoning compared to another dataset, even when both are encoded against one and the same ontology; sparse versus dense datasets; presence and size of literals; number of predicates used; usage of `owl:sameAs`, and other alignment primitives.
- *Hardware and software setup* – the performance can vary considerably depending on the version and configuration of the compiler or the virtual machine, the operating system, the configuration of the engine itself, and the hardware configuration, of course.

The size of a dataset loaded in a semantic repository is measured in different ways. These are related to the types of statements taken into consideration. The following measures should be considered:

- *Number of inserted statements* (NIS): How many statements have been inserted in the repository.
- *Number of stored statements* (NSS): How many different statements have been stored and indexed by the repository. For engines using forward-chaining and materialization, the volume of the data to be indexed includes the inferred triples. For instance, the RDF/OWL representation of WordNet expands after materialization from 1.9 million (NIS) statements to 7.1 million (NSS). In the opposite direction, NSS can be smaller than NIS, when one and the same statement is inserted multiple times.
- *Number of retrievable statements* (NRS): How many different statements can be retrieved from the repository. This number can be different from NSS when the repository supports some sort of backward-chaining. For instance, BigOWLIM performs `owl:sameAs` optimization, which reduces considerably the NSS in the repository.

7.3.4 Full-Cycle Benchmarking

The performances of a specific setup of a given semantic repository on loading datasets and query evaluation are interdependent. More comprehensive indexing and forward-chaining take time during loading and respectively make the loading performance worse in order to facilitate faster query processing. In fact, if query evaluation performance is not to be considered, loading of RDF could be as fast as the file-system could be in storing the input files locally with no overheads to maintain indices.

Here the notion of *full-cycle benchmarking* is introduced – an evaluation experiment, which provides a complete picture of the performance of a repository with respect to the full “life cycle” of the data within the engine. At the high-level, this means the measuring and publication of data for both loading and query evaluation performance within a single experiment or benchmark run. Thus, full-cycle benchmarking requires loading

performance data to be matched with query evaluation data. A full-cycle run on LUBM, [22], or similar benchmark usually involves the following activities:

1. Loading input RDF files from the storage system
2. Parsing the RDF files
3. Indexing and storing the triples
4. Forward-chaining and materialization (optional)
5. Query parsing
6. Query optimization
 - a. Query rewriting (optional)
7. Query evaluation, involving
 - a. Backward-chaining (optional)
 - b. Fetching of the results

While different repositories can employ different indexing, reasoning, and query evaluation strategies, the seven activities listed above should always be handled in a cycle that includes data loading and query evaluation.

To provide correct answers to queries, a semantic repository should consider the semantics of the data. For instance, it is impossible to deliver correct results to the queries of LUBM without some form of reasoning. As discussed in [Sect. 7.2.1](#), different reasoning strategies can be adopted. Inference can take place either during loading (step 4 above) or during query evaluation (steps 6a and/or 7a above).

Full-cycle benchmarking provides an adequate picture on the performance of the engine and its utility in real-world applications. It shows the implications of the different design choices and implementation approaches. This notion needs further extension to include data modification.

7.4 Performance Considerations and Distribution Approaches

This section covers several issues related to the implementation of efficient and scalable semantic repositories. It includes a discussion on the principal advantages and disadvantages of some distribution approaches along with practical aspects such as the current state of the server hardware and the typical sizes of the datasets used today. The latter provides a good perspective toward the economic efficiency of the different distribution approaches.

7.4.1 Performance Engineering and Hardware Trends

As stated earlier, semantic repositories represent a sort of database management systems (DBMS) and have performance requirements similar to those of the relational DBMS and the NoSQL solutions, [47], used as back-ends for some of the largest websites. The basic

requirement is fast random access to relatively small blocks of data (few kilobytes), retrieved from large volumes of data – the indices maintained by the engine to enable the fast retrieval of data records based on specific constraints or retrieval patterns. Thus, the major factors for the performance of a server, when running a semantic repository, are the amount of the available RAM and the random seek speed of the hard drives. In an ideal situation, all the data can be cached in the RAM, which provides 3 orders of magnitude faster random access time than enterprise class hard drives (few microseconds versus few milliseconds).

A sample server configuration is provided below that, considering the above high-level requirements and their experience, is appropriate for handling serious query loads against datasets comparable to the volume of several of the central linked data datasets (see [▶ Sect. 7.1.4](#)). While the needs of each usage scenario are different, this configuration could be used as a starting point for the estimation of the hardware requirements of a specific application.

7.4.1.1 Usage Scenario: 1 Million Queries per Day Against 10 Billion Statements

The scenario described here assumes that an enterprise is interested to provide, through its website, some sort of public information access service, which exposes proprietary content and data, which is annotated and interlinked with several linked data datasets. This could be the example of a media company, annotating its news with respect to a dataset like FactForge, described in [▶ Sect. 7.8.1.1](#), or a pharmaceutical company interlinking its internal data and clinical trial reports to resource like LinkedLifeData, [▶ Sect. 7.8.1.2](#). Such a scenario can be described as follows:

- Data size: between one and ten billion statements.
- Semantics: OWL 2 RL lightweight reasoning.
- Query loads: about one million queries per day; 10–20 queries per second. This load is comparable to the peak load at BBC’s World Cup 2010 website (see [▶ Sect. 7.8.2](#)).
- Update rate: about a hundred small updates per hour. Small updates can be described like transactions, which add or delete up to 1,000 statements, without changing the schemas and the ontologies. (Changes in the schemas are possible in such scenarios, and they can be adopted far quicker in semantic repositories as compared to relational databases. Still the complexity of such updates varies considerably and is hard for quantification.) Such updates are cases like updating the descriptions of several related entities or the metadata of several articles or pictures.

7.4.1.2 The \$10,000 Database Server Landmark

Considering the scenario described in the previous section using the benchmarking data and experience, provided below is a sample description of a server configuration which,

given the current situation on the hardware market, combines good efficiency with low cost per transaction per second. The server configuration with an assembly cost below \$10,000 is:

- 2 × Xeon 5500 series CPUs, each with four cores with hyper-threading
- 48–72 GB of RAM
- 8 × 120 GB SSD MLC drives with appropriate RAID controller, setup in RAID-0; those are to be used as fast storage for the indices of the semantic repository
- 2 × 2 TB SATA HDD set up in mirror (RAID-1), to be used for complementary data and content and for backup of the data placed on the SSD drives

The above configuration represents a commodity databases server, with the specificity that it is loaded with as much RAM as the configuration allows. This amount of memory would allow most of the engines to keep in memory several critical types of information (e.g., URL-to-internal-ID dictionaries) for datasets of a size up to 10–20 billions of statements. The solid state drives (SSD) and the hard disk drives (HDD) should be carefully selected to deliver best performance and reliability for this class of products.

More powerful database servers with four CPUs and 128 to 144 GB of RAM can be purchased for approximately \$20,000–30,000. While the price of such servers is disproportionately higher, compared to their capacity, they are still very likely to offer a more efficient solution for data scalability issues, compared to distribution approaches involving data partitioning (see [Sect. 7.4.2.1](#)). Such machines can handle up to 40–50 billion explicit statements and query loads approaching one thousand queries per minute.

One should consider that the above configurations represent an efficient solution in the year 2010. As the hardware evolves rapidly, they should be considered only as a reference point to estimate the required hardware for specific loads.

7.4.2 Distributed Semantic Repositories

The restriction to a single computer system limits the overall scalability and performance parameters of a semantic repository. Database management distribution is usually considered to address one or more of the following goals:

1. To handle efficiently larger volumes of data (The size of the data, managed by a DBMS, and the average time for the execution of specific operations with the data are always in dependency – thus the first point above can be seen as a generalization of the following ones. Data scalability issues can always be avoided by adoption of a DBMS or a DBMS configuration, which has lower hardware requirements, trading scale for efficiency. Still, the notion of “data scalability” appears here, because often change of the DBMS configuration or further lowering the hardware requirements is impractical.)
2. To speed up the data loading and indexing and to improve the performance for updates

3. To lower the query evaluation time for complex queries (e.g., analytical Business Intelligence reports)
4. To better handle concurrent query loads and large numbers of users
5. To ensure failover, for example, to surmount failure of one or more nodes and repositories

The general approaches for the distribution of database management systems are:

- *Data partitioning*, where the information stored and accessed by the system is spread across multiple machines, so that none of them contains the entire dataset
- *Data replication*, where the entire dataset resides on each of the machines

The remainder of this section provides discussion on the different approaches, their advantages and disadvantages, and appropriateness with respect to different scenarios and goals.

7.4.2.1 Data Partitioning

While data partitioning looks as the more promising schema, it is also the one which is most problematic to implement. In general, it enables the management of larger volumes of data and provides more space for in-memory data structures. Each node can apply more efficient caching and optimization with respect to the fraction of the data that it deals with. Data partitioning with redundancy also allows for failover support. Still, the major issue is that query evaluation against distributed data requires intensive communication between the nodes for exchanging intermediate results; the most common variety of such communication is known as “remote join.” Query optimization schemas, which consider the communication costs, are far harder to implement, which triggers less-optimal query evaluation plans and larger overall numbers of index lookups. In large number of scenarios, these effects neutralize the gains from the additional computing power gained from several machines. As outlined in [Sect. 7.2.1.1](#), the same concerns are the application of rule-based reasoning in repositories using data partitioning.

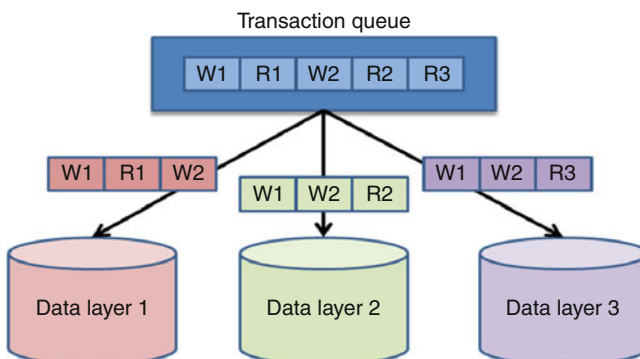
Two approaches to data partitioning appear in database systems from the established distributed DBMS research: horizontal and vertical partitioning. The horizontal data partitioning approach partitions a dataset across several repository nodes where no schema limitations apply to any of the nodes. A vertical partitioning approach would assign different parts of the data schema to different repository nodes so that later on requests for any type of data would be redirected to the respective repository node. This approach can be further extended and types of data that usually appear “close” together can be placed within a single node (when possible). In principle, such clustering would allow for whole sub-queries to be executed within a single node. It would therefore avoid the transfer of intermediate results between the repository nodes and the central query processing node only to complete the query. Overall, one should consider that data partitioning comes together with problems which are not trivial to solve and imply

compromises in the engineering design. The CAP theorem [9], summarizes the results of significant amount of theoretical and experimental work in distributed database systems by stating the trade-offs between consistency, availability, and partition-tolerance. Hybrid schemes are possible for datasets that are both horizontally and vertically too large to fit in a single computational node. Such a partitioning scheme, however, would complicate considerably the query processing subsystem, which would have to take into account splits and joins in both dimensions while planning the query execution.

7.4.2.2 Data Replication

Data replication is a traditional approach for boosting the read performance of a DBMS at the cost of redundancy and write propagation complexity. In a classic scenario, several slave nodes are assigned incoming read requests by a central master node that performs any sort of load balancing (e.g., round robin) to distribute the load evenly across the slaves. Writes are executed on the master node, and updates are propagated to the slaves in the background. Such a setup is very appropriate in situations when a lot of read requests occur while write requests are rare or clustered together in large batches (for example, if a large dataset is initially loaded in the repository). In such situations, the resource-intensive replication procedure will not be necessary most of the time, while a theoretically linear scalability will take place on the reading side.

In an alternative replication implementation, write operations are propagated to all the slave nodes which operate as independent repositories. Such schema is implemented in BigOWLIM (see [Sect. 7.6.4](#)) and presented on [Fig. 7.7](#). In this case, the total loading and modification performance of the cluster is equal to that of the fastest slave node. The data scalability of the cluster is bounded by those of the weakest slave node. Such design, however, simplifies the role of the mater node and the communication within the cluster, bringing principal advantages with respect to the resilience of the cluster.



■ Fig. 7.7

Data replication with propagation of write operations

In case of failure of one or more instances, the performance degradation is graceful and the cluster will be fully operational even when there is only one instance working.

Using replication excludes the remote join problem, as each node can evaluate the query without exchanging information with the other nodes. As a result, there are no query performance issues, which facilitates a better handling of concurrent user requests. Load balancing and failover are easy to implement. The cluster can scale horizontally to handle virtually unlimited query loads, as the query evaluation capacity is a sum of the capacities of all the slave nodes in the cluster. Still, replication does not deliver better scalability with respect to larger datasets and the usage of resources is not optimal because all the loading and modifications need to be performed on all nodes.

7.4.2.3 Advantages of the Different Distribution Approaches

As an overview of the two major distribution approaches we can summarize that:

- Data partitioning improves data scalability; however, in most of the cases it hampers the query evaluation performance due to high communication overheads. It can improve loading performance if there is no materialization involved (see [Sect. 7.2.1.2](#)).
- Data replication allows for a better handling of concurrent query loads and failover. It is neutral with respect to loading and inference performance.

None of these approaches provide a principal advantage for the evaluation of complex queries. Under data replication, one of the nodes can be off-loaded from concurrent queries, which would allow faster execution of a complex query. An approach known as “federated join” can in theory improve the performance of such queries in very specific data partitioning scenarios, where the communication costs can be minimized.

As already mentioned, a direct consequence of the CAP theorem [9] is that data consistency might have to be sacrificed if availability and partitioning are required. One might argue that data inconsistency could be a serious issue in certain types of real-world production-level applications. An eventually consistent and highly scalable distributed system could be very well suited for the semantic repository of an incomplete reasoning system, where result completeness is not a requirement by design.

7.4.3 Multi-Core Parallelism

In addition to the various multi-node data distribution approaches, parallelization at thread level within a single computer system can also be considered. This is the approach of multi-core parallelism. These days even commodity desktop hardware is equipped with dual or even quad-core CPUs that are better utilized by multi-threaded implementations whenever these are appropriate. The cost-efficient DB servers (as the one presented in [Sect. 7.4.1.2](#)) come with 8–12 CPU cores, each of which is an autonomous computing device, while all cores share the same RAM. A multi-threaded semantic repository can benefit from parallel execution using multiple CPUs for computation, as with the

repositories distributed across several machines, but without the communication costs and overheads of the distributed approach, in particular without the remote join problem. One should consider that efficient multi-threading with respect to loading and modification requires nontrivial locking and synchronization, whereas multi-threaded handling of read-only operations (like query evaluation) is straightforward. One possible candidate for multi-threading is the “federated join” approach in which the outer-most loop of the query evaluation routine (i.e., the one traversing the outer-most triple pattern of the query) is split among several threads that operate over equal parts of the repository. This is another case that would be appropriate in scenarios when reads occur a lot and writes are rare.

Both Virtuoso and BigOWLIM support multi-threading and can efficiently use multi-core CPU as it becomes evident from the results of the BSBM benchmark presented in [Sect. 7.2.1](#). As anticipated, multi-threaded query evaluation works without extra effort. The query throughput grows until the number of simultaneous users reaches the hardware support for parallelism. However, the usage of multi-threading for loading and inference requires special configuration. Local data partitioning is applied to minimize the interlocking between multiple threads used for loading and inference.

7.5 Popular Benchmarks and Datasets

The data stored in the semantic repositories form datasets. Their most distinctive features are their size and their complexity with respect to reasoning and querying. Benchmarks can be created through the combination of datasets and predefined sets of queries, which along with the datasets are commonly used as measuring sticks for evaluating the performance of the semantic repositories. A few of the most popular are presented here.

7.5.1 Lehigh University Benchmark (LUBM) and UOBM

The most popular benchmark for semantic repositories with support for RDF and OWL is Lehigh University Benchmark (LUBM), defined in [22]. The purpose of the benchmark is to measure the performance of storing and querying of large amounts of data that are created for realistic Semantic Web systems. It employs synthetically generated datasets using a fixed OWL ontology of university organizations, lecturers, teachers, students, and courses. The complexity of the language constructs used is between OWL Horst and OWL DLP [17], [50]. Fourteen queries are defined that are used to check the query evaluation correctness and speed of repositories that have loaded a given dataset. The biggest standard dataset is LUBM(50) (i.e., it contains synthetic data for 50 universities). Its size is 6.8 million explicit statements, distributed in 1,000 RDF/XML files with total size of 600 megabytes. For the purposes of scalability measurements, many groups have used the LUBM generator to create bigger datasets.

This is the dataset which is adapted practically by all major semantic repository vendors. Through the years, it played a considerable role in the development of the

field. Many of the scalable reasoning records were generated on the basis of LUBM – in 2009, 5 years after LUBM was published, BigOWLIM loaded a dataset of 90,000 universities. The major critiques against LUBM are as follows:

- The RDF graphs generated are very easy for partitioning and could provide misleading positive results for some query evaluation, reasoning, and distribution approaches that otherwise may not perform well on top of real-world datasets such as UNIPROT and DBPedia (described in the next sections).
- Some of the queries (most notably Q2 and Q9) return results, which are proportional to the size of the dataset, which for very large datasets is impractical and introduces distortion of the query evaluation results.

UOBM [40], is a benchmark that puts on test scalable ABox (instance data) reasoning against a relatively inexpressive DL ontology. UOBM is a further development of the LUBM benchmark; it uses the same evaluation framework (i.e., Java libraries), but provides alternative ontology, knowledge base, and queries, which allow for:

- More comprehensive coverage and usage of the OWL Lite and OWL DL semantics.
- Additional connections across the datasets for different universities – this modification assures higher level of connectivity in the RDF graph, which provides a more realistic and interesting test case.

The UOBM benchmark includes two distinct datasets for OWL Lite and OWL DL, each of which includes data collections for one, five, or ten universities, named respectively: Lite-1, Lite-5, Lite-10, DL-1, DL-5, and DL-10. No dataset generator for UOBM is publicly available at present. The Lite version of the test contains 13 queries; the DL version adds two more. Both the Lite and the DL variants of UOBM require considerably more complex reasoning to be performed by a semantic repository in order to provide correct answers to the queries.

7.5.2 UniProt

UniProt (Universal Protein Resource, <http://www.uniprot.org>) is the world's most comprehensive and most popular dataset of information on proteins, created by joining several other resources (Swiss-Prot, TrEMBL, and PIR). UniProt RDF, [59], is an RDF representation of the dataset; it is based on an OWL ontology, expressed in a sub-language of OWL Lite, that is more expressive than OWL Horst, but still tractable (see ▶ Sect. 7.2.2 for information on the different OWL dialects). It represents one of the largest datasets distributed in RDF and OWL. Processing UniProt is often used as benchmark for scalability and reasoning capabilities of semantic repositories. Still, very few of the repositories are capable of loading UniProt and perform materialization on top of it or to interpret its semantics otherwise.

The size of the RDF representation of UniProt is above 1.5 billion unique explicit statements. The RDF graph defined in the UniProt ontology is highly interconnected,

which has significant impact on the loading and reasoning speed of the semantic repositories. Today UniProt is one of the central datasets in the biomedical part of the Linking Open Data (LOD) initiative [38], [39].

The RDF representation of UniProt was used for benchmarking Jena and AllegroGraph (see ▶ Sects. 7.6.6 and ▶ 7.6.2). It is also part of the Pathway and Interaction Knowledge Base (PIKB), [2], and the LinkedLifeData service presented in ▶ Sect. 7.8.1.2.

7.5.3 DBPedia and Other Linked Datasets

DBPedia (More information about DBPedia is provided by its developers in ▶ [Semantic Annotation and Retrieval: Web of Data](#)) is a dataset represented in RDF the Infobox of Wikipedia together with other information related to or derived from Wikipedia. It serves as a connectivity hub of the Linked Open Data (LOD) initiative, [39]. The diversity of the information represented in DBPedia and the fact that it represents encyclopedic knowledge make it an excellent resource for benchmarking semantic repositories. DBPedia version 3.3 includes 362 million unique statements without the `owl:sameAs` links.

FactForge (▶ Sect. 7.8.1.1) is probably the largest and most heterogeneous body of general factual knowledge that was ever used for logical inference. It integrates in a single repository several of the most central datasets of Linking Open Data (LOD) cloud [39]. When used for benchmarking of semantic repositories, it has several advantages, compared to the straightforward usage of DBPedia:

- It is more diverse, as it represents data and data modeling patterns from several other datasets as well. One should consider for instance that DBPedia and Geonames are datasets of a very different nature. A repository, which performs well on Geonames could show poor performance when dealing with DBPedia for various reasons; one of them being that Geonames uses few tens of different predicates, whereas DBPedia uses around a hundred thousand predicates.
- The data in FactForge can be used for inference, because they were cleaned up and prepared to allow this. On the other hand, reasoning with the raw DBPedia data is inappropriate as it results in the inference of a very large number of false and useless statements; the latter happens mostly due to problems with the category hierarchy discussed in [34].

7.5.4 Berlin SPARQL Benchmark (BSBM)

Berlin SPARQL Benchmark, [6], evaluates the performance of query engines in an eCommerce use case: searching products and navigating through related information. Randomized “query mixes” (of 25 queries each) are evaluated continuously through a client application that communicates with the repository through a SPARQL endpoint.

The benchmark enables evaluation with respect to the changing sizes of the dataset and differing numbers of simultaneous clients.

Although created for the benchmarking of SPARQL engines, the design of BSBM favors relational databases and other raw-store-based implementations, as long as they deal with a single fixed data schema and uniform dense data representation. Generally, RDF databases are designed to deal efficiently with diverse data, integrated from multiple data sources, encoded against different data schema, resulting in sparse data tables in relational databases (see [▶ Sect. 7.1.2](#)).

BSBM supports the benchmarking of relational engines, as there is an SQL-based version of the dataset and the queries. One should note that the semantics of some of the queries in the SQL version is simpler than those of their SPARQL equivalents, that is, the SQL versions are less powerful and return different results. The evaluation results published in [7] prove that relational databases are really more suitable for such loads – considering that the results are not truly comparable, it is still amazing that the relational engines are one order of magnitude faster.

Finally, unlike LUBM ([▶ Sect. 7.5.1](#)), BSBM does not require any inference – an engine can deliver correct results of the queries without any interpretation of the semantics of the data. Still, the benchmark is useful for evaluation of the SPARQL support of the engines and their efficiency in handling multi-client loads.

7.6 Semantic Repository Engines

Several of today’s outstanding semantic repositories are presented in alphabetical order in this section with discussion on their specific advantages and features.

7.6.1 3store, 4store, 5store

3store, 4store, and 5store represent a family of RDF database engines developed by Garlik (<http://www.garlik.com/>) – a company dealing with online identity and personal information protection in UK. 4store (<http://www.4store.org>) is an open-source RDF storage engine designed to run in cluster setup of up to 32 nodes. 4store implements a distribution schema based on data partitioning (see [▶ Sect. 7.4.2.1](#)); more details about it are available in [23]. While 4store does not offer inference capabilities, [57] presents the 4s-reasoner, which implements RDF reasoning through backward-chaining on top of 4store.

It is implemented in ANSI C99 and is available for UNIX-based operating systems only, including Linux and MacOS. 4store is advertised with the fact that at Garlik it is “holding and running queries over databases of 15GT, supporting a Web application used by thousands of people.” The RDF repositories created with 4store are managed through a set of command line utilities that allow the importing of RDF data in various formats,

querying using SPARQL, backup, and other maintenance tasks. The 4store distribution also includes a stand-alone SPARQL HTTP protocol server as a standard interface to the 4store repository.

As the names suggest, 3store is the predecessor and 5store is the successor of 4store. According to the website of 5store (<http://4store.org/trac/wiki/5store>), it offers the same functionality and interfaces as 4store, but features a new architecture which allows for the handling of larger clusters and better performance. There is no public information about benchmarking 4store or 5store with respect to popular benchmark datasets; there are also no publications about independent performance evaluations of 4store and 5store.

7.6.2 AllegroGraph

AllegroGraph RDFStore, [1, 18], is an RDF database with support for “SPARQL, RDFS++, and Prolog reasoning from Java applications.” In addition to the “basic” DBMS functionality, it offers specific support for geo-spatial data handling and social network analysis.

AllegroGraph does not perform reasoning and materialization during loading. The so-called RDFS++ reasoning can be switched on during query processing. The semantics supported in this way is comparable to OWL Horst ([17]). RDFS++ reasoning allows AllegroGraph to deliver correct results on LUBM benchmark. Versions 3.2 and 4.0 of AllegroGraph support the so-called RDFS++ dynamic materialization, [20], which seems to require building some extra indices on-the-fly, whenever those are required for query evaluation. This approach helps AllegroGraph report excellent results, [19], on query evaluation in LUBM(8000).

One of the major developments in AllegroGraph 3.0 is the “federation” that enables the grouping multiple stores (running locally or on a remote machine) within a single virtual store. Federation has the potential to considerably speed up the loading process, as the work is effectively distributed across multiple stores. The distribution approach implemented in AllegroGraph is based on data partitioning (see [Sect. 7.4.2.1](#)).

7.6.3 BigData

BigData (<http://www.systap.com/bigdata.htm>) is an open-source distributed B + Tree database, designed to accommodate large RDF repositories and scale horizontally on commodity hardware. Scaling is achieved by index partitioning across the nodes of the cluster. As the data evolve, partitions of the indices might get split, moved, or joined transparently and asynchronously to the client. A centralized metadata index takes care of locating the index partitions in the cluster. As discussed in [60] and [61], BigData is

designed to support various modalities of transactional isolation through multi-version concurrency control (MVCC, [55]) and aims to support load balancing internally.

BigData integrates with the Sesame 2.0 platform (see [Sect. 7.6.8](#)) to achieve SPARQL support and therefore relies on it for query parsing but implements query optimization based on the fast key-range counts supported natively by the B + Tree architecture.

BigData supports a sort of RDFS + inferencing backed by a hybrid approach of materializing the entailments of some rules (forward-chaining) at load time and using backward-chaining for others at query time. Other features of BigData include statement-level provenance and full-text indexing.

7.6.4 BigOWLIM

OWLIM is a semantic repository implemented in Java and packaged as a Storage and Inference Layer (SAIL) for the Sesame RDF database (see [Sect. 7.6.8](#)). OWLIM is based on TRREE – a native RDF rule-entailment engine. The standard inference strategy is forward-chaining and materialization. The supported semantics can be configured through rule-set definition and selection. The most expressive predefined rule-set is OWL 2 RL, [45]. Custom rule-sets allow tuning for optimal performance and expressiveness.

The two major varieties of OWLIM are SwiftOWLIM and BigOWLIM. In SwiftOWLIM, reasoning and query evaluation are performed in memory, while, at the same time, a reliable persistence strategy assures data preservation, consistency, and integrity. BigOWLIM is the “enterprise” variety that deals with data and indices directly from disk or other file storage, which allows it to scale more comprehensively. Further, BigOWLIM’s indices are specially designed to allow efficient query evaluation against huge volumes of data. SwiftOWLIM can manage millions of explicit statements on desktop hardware. Given an entry-level server, BigOWLIM can handle billions of statements and serve multiple simultaneous user sessions.

The most advanced features of version BigOWLIM 3.3, released in June 2010 and used in BBC’s World Cup 2010 website (see [Sect. 7.8.2](#)), can be summarized as follows:

- Pure Java implementation compliant with Sesame (see [Sect. 7.6.8](#)). The latter brings interoperability benefits and support for all popular RDF syntaxes and query languages, including SPARQL
- Clustering support brings resilience, failover, and horizontally scalable parallel query processing (see [Sect. 7.4.2.2](#))
- Optimized handling of `owl:sameAs` (see [Sect. 7.2.1.5](#)), which delivers considerable improvements in performance and usability when huge volumes of data from multiple sources are integrated
- Full-text search, based on either Lucene or proprietary techniques
- High-performance retraction of statements and their inferences (see [Sect. 7.2.1.6](#))

- Logical consistency checking mechanisms, as those supported in OWL 2 RL and OWL Horst (see [Sect. 7.2.2](#))
- RDF rank, similar to Google’s PageRank, can be calculated for the nodes in an RDF graph and used for ordering query results by relevance
- A notification mechanism, to allow clients to react to updates in the data stream

A feature of BigOWLIM 3.3 which deserves special attention is the so-called RDF Search, which provides a novel method for the schema-agnostic retrieval of data from RDF datasets. The main rationale behind RDF search is to allow one to search in an RDF graph by keywords and get usable results (stand-alone literals are not useful in many cases). Technically, it involves the full-text indexing of the URIs in the RDF graph with respect to their “text molecules” – text snippet achieved by means of the concatenation of text from all the nodes in the RDF-molecule of the corresponding URI. The result is list of URIs, ranked with a metric combining the standard full-text search Vector Space Model and the RDFRank. In FactForge (see [Sect. 7.8.1.1](#)), each of the URIs in the result list is presented with human-readable labels and text snippets.

7.6.5 DAML DB, Asio Parliament

DAML DB is an older name of the engine of BBN Technologies, which currently appears as part of Parliament (<http://www.bbn.com/technology/knowledge/parliament>) – an open-source knowledge base management system that implements a high-performance storage engine, compatible with the RDF and OWL standards. It is usually combined with query processing frameworks, such as Sesame ([Sect. 7.6.8](#)) or Jena ([Sect. 7.6.6](#)), to implement a complete data management solution with support for SPARQL query language.

Although there are no recent evaluation results published, DAML DB still seems to be one of the very few systems that can demonstrate a full-cycle benchmark results (see [Sect. 7.3.4](#)) on test like LUBM(8000).

7.6.6 Jena TDB

TDB (<http://jena.hpl.hp.com/wiki/TDB>) is an open-source RDF storage layer for the Jena Semantic Web Java framework (<http://jena.sourceforge.net/>). It is implemented purely in Java and can be accessed through Jena APIs or through several separately provided command line scripts. Paired with Jena, TDB provides a single-machine, non-transactional RDF storage and query environment that can be accessed over the SPARQL protocol when running inside the Jena-based Joseki HTTP server. SPARQL queries over TDB are made possible through Jena’s ARQ SPARQL query engine.

A TDB repository can be operated by 32-bit and 64-bit JVM without any format migration being necessary (a direct consequence of Java’s architecture independent types).

However, in 64-bit mode, TDB uses memory-mapped files to access its repository binary representation (data and indices). This is reported to contribute to a much better performance compared to the 32-bit case where data caching is handled by the TDB engine itself. Another benefit from relying on the operating system to do the file caching is the dynamically determined amount of memory used by the engine for disk caches.

TDB is equipped with different SPARQL query optimizers (e.g., fixed and statistical) that aim to optimize SPARQL queries on a per-repository basis. The fixed optimizer only considers the number of variables in a query's triple pattern in its reordering decision. The statistical optimizer relies on rules that approximate the number of results to be expected from a triple pattern (those rules could be either written manually or generated by the engine). TDB interprets RDF simple types (e.g., `xsd:integer`), which makes it possible to optimize SPARQL range filters.

7.6.7 ORACLE

ORACLE offers RDF support as part of the Spatial option of its DBMS since version 10 g R2. As presented in [69], in version 11 g R2, this support is improved in several ways, including:

- Support for the OWL Prime dialect, which is comparable with the owl-max semantics of OWLIM, [50]. The Pellet DL reasoner is integrated for T-Box (schema level) inference.
- The efficiency of RDF loading and inference is considerably improved.

ORACLE supports RDF models with named graphs, that is, it can be seen as a quadruple store. The semantics to be used for inference is defined in terms of rule-bases, which essentially represent sets of entailment rules. Inference is implemented in terms of forward-chaining and materialization – the results are stored in rule indices. The initiative regarding inference is given to the user, who should take care of it explicitly:

- Force inference, that is, generation of the rule indices.
- Invalidate the rule indices when necessary, that is, after statements are deleted.

The latest results from benchmarking ORACLE 11 g are published in [65]. One should consider summing up the times for the different steps. For instance, the times reported for LUBM(8000) are as follows:

- Bulk-load: 30 h. 43 min.
- Loading into staging table: 11 h. 32 min. (when proper correctness checks are performed by the RDF parser).
- Inference (OWL Prime): 11 h. Multi-threaded inference runs 3.3 times faster as compared to single-threaded one on a quad-core CPU.

Two important circumstances should be acknowledged:

- The LUBM(8000) results for ORACLE are measured on desktop computer, while most of the other results are measured on servers. In fact, the configuration used for inference is comparable to the workstation used for benchmarking BigOWLIM on LUBM(8000).
- ORACLE is also the most “economic” with respect to the RAM usage – the above results reflect the inference run with 4 GB of RAM, but results measured on 2 GB systems suggest graceful degradation of the performance when less memory is available.
- The results reported for loading and for inference come from different machines; there are no public results for loading times at the same CPU where ORACLE achieves its best inference time.

7.6.8 Sesame

Sesame is one of the most popular semantic repositories that supports RDF(S) and all the major syntaxes and query languages related to it. Sesame is ranked by multiple independent evaluations (e.g., [53]) as the most efficient RDF repository framework. Several engines rely on the Sesame RDF database framework (<http://www.openrdf.org>). Semantic repositories like OWLIM and BigData (see ▶ Sects. 7.6.3 and ▶ 7.6.4) use Sesame as a library, taking advantage of its APIs for storage and querying, as well as the support for a wide variety of query languages (e.g., SPARQL and SeRQL) and RDF syntaxes (e.g., RDF/XML, N3, Turtle). Other engines like Virtuoso and AllegroGraph use it just for the sake of interoperability.

7.6.9 Virtuoso

OpenLink Virtuoso (<http://www.openlinksw.com/virtuoso/>) is a “universal server” offering diverse data and metadata management facilities, for example, XML management, RDBMS integration, full-text indexing, etc. The core engine of Virtuoso is a relational database engine with numerous RDF-oriented adaptations in datatypes, index layout, and query optimization.

Virtuoso does not have built-in materialization of entailment during loading. Instead, it supports the semantics related to subclasses, sub-properties, and owl:sameAs at runtime, through backward-chaining on the basis of a specified RDF schema. Thus, the user is not required to rewrite queries. Materialization is possible by writing SPARUL statements to generate implied triples.

Virtuoso uses bitmap indices with key compression to address the issue of space efficiency, and samples the database at query optimization time by keeping the data in memory to address the issue of speed of processing. It is designed with partitioning, specified at the index level – a hash partitioning where the hash picks a logical partition out of a space

of n logical partitions, where n is a number several times larger than the expected maximum machine count. Each logical partition is then assigned to a physical machine. When loading RDF data, the database translates between IRI's and literals and their internal identifiers. It then inserts the resulting quad in each index. An RDF query consists of a single key lookups grouped in nested loop joins with occasional bitmap intersections. The basic query is a pipeline of steps where most steps are individually partitioned operations. The partitioned pipe function may return a set of follow-up functions and their arguments, which get partitioned and dispatched in turn. In Virtuoso, each logical partition is allocated on multiple nodes. At query time, a randomly selected partition is used to answer the query if the data are not local. At update time, all copies are updated in the same transaction. This is replicated on all nodes. When loading data at a rate of 40Ktriples/s, the network traffic is 170 messages/s and the aggregate throughput is 10 MB/s with no real network congestion. Scale can be increased without hitting a network bottleneck. Thus Virtuoso can run complex queries with a reasonable number of messages, about $1620/34 = 47$ messages per query. As data are becoming a utility, the objective of Virtuoso and semantic repositories in general is to provide for the rapid deployment of arbitrary scale RDF and other database systems for the clouds. This involves automatic partitioning and re-partitioning and adapting query planning cost models to data that contain increasing inference.

The developers of Virtuoso report in [14] the results of various “distributions of labor” between materialization and query expansion. As expected, it appears that extensive materialization can save a lot of computations during querying, while the same query completeness is achieved. In principle, this type of inference can be implemented on top of any DBMS – the complexity of the inference depends on the complexity of the query language supported. Here the semantics is not enforced by the engine, but it is rather a matter of handcrafted queries that may or may not correctly reflect the semantics of the ontology language primitives.

The data provided in [14] leave several questions open:

- The queries of LUBM are modified, so it is not clear: (1) how the query evaluation performance compares with that of other engines benchmarked with the original LUBM queries and (2) whether the query results are truly complete and whether the implemented inference mechanisms are correct and sufficient for a full LUBM run.
- No performance data are provided for entailment and materialization, that is, there are no indications about the performance and efficiency of the various inference configurations that were put on test.
- Implementing the semantics of transitive properties via query-based entailment and materialization, as presented in [9], requires recursive query evaluation. However, there are no comments on the implementation of such behavior in Virtuoso.

Version 6.0 is the latest generation of the Virtuoso engine, which supports distributed RDF database management. The results (<http://www.openlinksw.com/dataspace/vdb/weblog/vdb%27s%20BLOG%20%5B136%5D/1563>) from the loading of LUBM(8000) in a cluster of eight instances running on a single dual-CPU, eight-core, server represent the fastest load on this benchmark on a server worth less than \$10,000.

7.7 State of the Art in Performance and Scalability

This section provides a discussion on the current state of the art with respect to scalability and the different aspects of the performance (outlined in ▶ Sect. 7.3). Reference [33] presents an overview and comparison of the most relevant publicly available benchmark results from several of the most prominent semantic repositories. Such a comparison is not presented here because in the year 2010, the rate of publication of new results is already quite high, which means that any specific results will be out of date soon. Further, the results being published are very hard to compare in a reliable manner for several reasons: Most of the vendors do not publish sufficient information about the benchmarking experiments; most of the information published does not comply with the full-cycle benchmarking principles (presented in ▶ Sect. 7.3.4); the hardware used by the different vendors is very different in terms of both price and performance. Instead, the remainder of this section generalizes some trends and refers to reports on independent evaluations and comparisons of semantic repositories, such as [63] and [8].

7.7.1 Data Loading Performance

The current publications at the “Large Triple Stores” page (a wiki page maintained by W3C at <http://esw.w3.org/LargeTripleStores>, where vendors report on their scalability achievements) suggest that there are several repositories which can load RDF datasets of a size between 10 and 20 billion statements. Most of the vendors report such results using servers similar to the one presented in ▶ Sect. 7.4.1.2. There are however certain differences:

- According to a post at the “Large Triple Store” (no further information is provided elsewhere), a Virtuoso (See ▶ Sect. 7.6.9 for more information about the engine) cluster of eight servers has loaded 15 billion statements of the LOD Cloud Cache (<http://lod.openlinksw.com/>) service. No average loading speed for the entire dataset is provided, here is a quote “*Latest bulk load added ~3 Billion triples in ~3 h — roughly 275Ktps (Kilotriples-per-second) — with partial parallelization of load.*” Materialization is not performed
- According to [33], BigOWLIM (See ▶ Sect. 7.6.4 for more information about the engine) uses a single server to load the 12 billion statements of LUBM(90,000), perform materialization, and index 20 billion statements. Two speeds should be considered here because of the materialization: 12,000 statements/s. is the speed of loading explicit statements; 18,000 statements/s. is the speed of indexing statements
- Again according to a post at the “Large Triple Store” with no further information provided elsewhere, 4Store (See ▶ Sect. 7.6.1 for more information about the engine) “. . . is running with 15B triples in a production cluster”. While no proper reference is provided for the average loading speed, values in the range of 140,000 statements per second are indicated. 4Store does not perform materialization.

To summarize, systems which do not perform materialization and feature data partitioning demonstrate speeds in the range of few hundred thousand statements per second. Still, to understand the importance of these achievements, one should be able to analyze also query evaluation correctness and performance. This is important in order to distinguish really comprehensive implementations, on the one end of the spectrum, from a trivially partitioned store (TPS) like the following one, on the other end. Suppose that TPS is a distributed repository which is implemented as a cluster of N nodes, each of which being a stand-alone semantic repository. New statements are given a hash code and distributed to one of the nodes; materialization is not supported. All low-level read requests are broadcasted to all the nodes, and the results are federated at the master node. Such a TPS will exhibit perfect horizontal scalability with respect to the volumes of data and the speed of loading; given a hash function which distributes the statements evenly, the loading speed of the TPS will be close to the sum of the speeds of all the nodes. However, the query evaluation speed of the TPS will be very poor due to the high communication costs.

Under full-cycle benchmarking conditions, the best loading speeds for datasets in the range of one billion statements are in the range of 100,000 statements per second; such is the case of AllegroGraph's multi-threaded loads, [19]. As expected, and as observed in the results comparison in [33], materialization with respect to OWL Horst-like languages, even for the synthetic datasets of LUBM (See [▶ Sect. 7.5.1](#)), brings the loading speed down to the levels of a few tens of thousands of statements. This is the case even with multi-threaded loading and inference implementations as the one of ORACLE, [69].

Probably the most interesting and useful performance analysis results are presented in [63] where several of the outstanding repositories are evaluated with respect to a real-world scenario. The repository is populated with: an ontology; a factual knowledge base of about seven million statements; and eight million statements of metadata about a collection of Press Association, containing five million images. Next 15 queries relevant to a content management and Web publishing are evaluated. The scenario requires interpretation of the ontology with respect to OWL Horst semantics. The test setup makes no assumptions whether reasoning is implemented through forward-chaining and materialization during loading or through backward-chaining during query evaluation. AllegroGraph loads the data about ten times faster than the engines which perform materialization (Sesame, Jena TDB, and BigOWLIM); as expected, this has implications on the query evaluation times presented in the next section.

7.7.2 Query Evaluation

Loading, combined with query evaluation, can deliver full-cycle benchmarking ([▶ Sect. 7.3.4](#)), i.e., an adequate picture of the performance of the engine. Below we provide publicly available results about query evaluation performance of the engines presented in [▶ Sect. 7.6](#) based on two benchmarks: BSBM ([▶ Sect. 7.5.4](#)) and LUBM ([▶ Sect. 7.5.1](#)). We will start, however, with the result of a query evaluation benchmarking

a real-world scenario of Press Association, reported in [63] and introduced in [Sect. 7.7.1](#). While AllegroGraph was much faster in the loading of the data, it fails to answer two of the queries and its average query evaluation time is 50 times slower than that of BigOWLIM (8.2 s. versus 0.16 s.) This is a clear demonstration of the trade-offs between the two main reasoning strategies (see [Sect. 7.2.1.2](#)).

7.7.2.1 BSBM Results

Recent results from evaluation of few of the most popular engines with the BSBM benchmark (see [Sect. 7.5.4](#)) are published in [7] and [8]. The former includes: relational database engines (running the SQL version of the benchmark), relation-to-RDF wrappers, and native RDF engines. As long as BSBM is biased toward relational databases, it is not a surprise that the relational engines (MS SQL and Virtuoso SQL) perform far better than the native RDF stores and the relational-to-RDF wrappers are somewhere in the middle ([Fig. 7.8](#)).

[Figure 7.9](#) provides a comparison between results from the internal evaluation of BigOWLIM 3.1 and results for other systems from [7] regarding query evaluation with a growing number of simultaneous clients (1, 4, 8) against 25-million statement version of the BSBM dataset. Unfortunately, there are no public BSBM results on the evaluation of engines, other than OWLIM, with more than four clients for datasets larger than 25 M. On the other hand, for the most mature engines, the indices of the 25 M dataset fit into the main memory of any machine with more than 4 GB of RAM. Still, the results demonstrate the degree to which the engines can parallelize query processing in a way which utilizes the hardware more efficiently.

Given a larger number of simultaneous clients, BigOWLIM and Virtuoso can deliver considerably larger overall throughput, almost proportional to the number of CPU cores of the system. BigOWLIM's results were acquired on a desktop (osol) with a quad-core CPU with hyper-threading support; Virtuoso's results were acquired on a desktop quad-core CPU without hyper-threading.

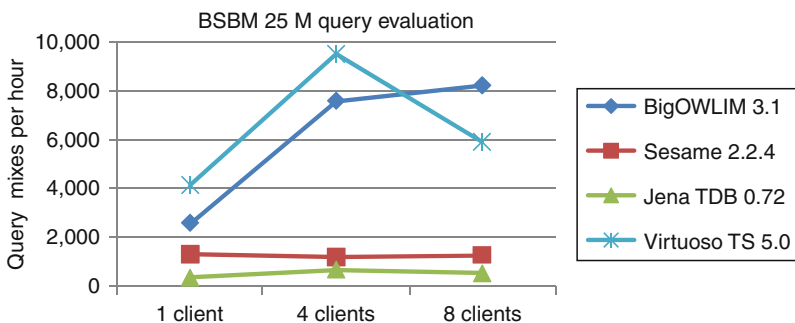
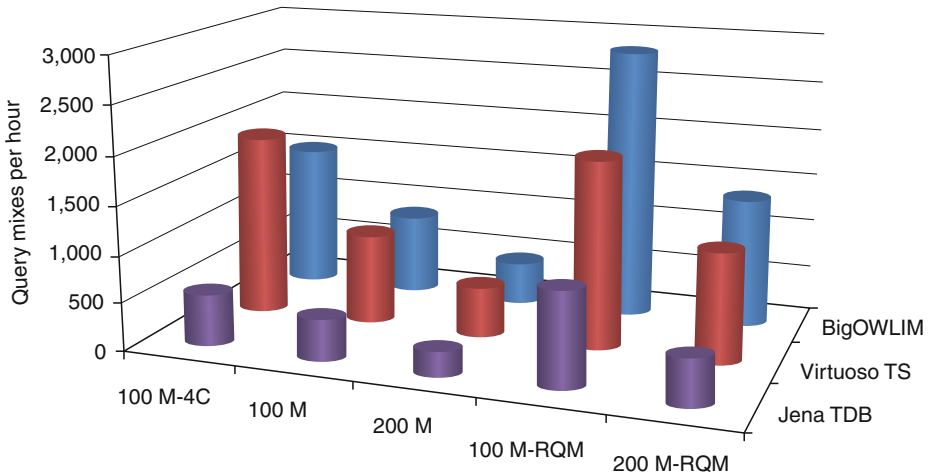


Fig. 7.8

BSBM query results: Multiple clients



■ Fig. 7.9

BSBM query results: Large datasets and reduced Query mix

In subsequent experiments, [8], the authors of BSBM evaluated the performance of BigOWLIM, Jena TDB, and the native RDF Virtuoso engine. Excerpts of these results are presented in [Fig. 7.9](#). The experiments included two scales: 100 M and 200 M. Apart from the standard runs, there are also results for two variants: running with four clients (indicated on the diagram with “4C”) and running a “reduced query mix” (indicated with RQM). The reduction of the query mix was made by removing a couple of queries, which are not suitable for benchmarks at scales above 25 M; at large scale, they consume above 80% of the query time for the entire mix. Analysis of these results follows:

- As already observed on [Fig. 7.9](#), Virtuoso and BigOWLIM are efficient in parallelizing the query evaluation.
- Both Virtuoso and BigOWLIM can handle about 10 nontrivial queries per second against 200 M dataset, even running in single-client mode on desktop machine worth less than \$1,000.

7.7.2.2 LUBM Results

Although there are plenty of data on loading performance and the scalability of LUBM, in a very few cases, query evaluation has been benchmarked within the same environment that was used for dataset loading.

The table below presents the results of query evaluation of the LUBM benchmark with 8,000 universities. Two sets of results are provided for BigOWLIM 3.1 – for server (onap) and desktop (osol). The results of AllegroGraph 3.2, [19], were acquired on a server comparable to the one used for BigOWLIM. The semantic repository engines in question are described in [Sect. 7.6](#), and specifications of the benchmark environments are given in [Sect. 7.7.1](#).

Query performance LUBM(8000), one billion statements

Query No	# of results	BigOWLIM 3.1 (onap)		BigOWLIM 3.1 (osol)		AllegroGraph 3.2	
		Time (msec)	QTPR (msec)	Time (msec)	QTPR (msec)	Time (msec)	QTPR (msec)
1	4	23	5.8	52	13.0	4	1.0
2	2,528	251,992	99.7	873,197	345.4	54,964	21.7
3	6	26	4.3	25	4.2	2	0.3
4	34	8	0.2	28	0.8	14	0.4
5	719	38	0.1	29	0.0	37	0.1
6	83,557,706	84,333	0.0	174,777	0.0	104,794	0.0
7	67	38	0.6	71	1.1	9	0.1
8	7,790	113	0.0	132	0.0	178	0.0
9	2,178,420	523,215	0.2	1,460,862	0.7	117,303	0.1
10	4	27	6.8	24	6.0	5	1.3
11	224	4	0.0	23	0.1	8	0.0
12	15	6	0.4	33	2.2	14	0.9
13	37,118	1,662	0.0	121,980	3.3	47	0.0
14	63,400,587	63,998	0.0	157,568	0.0	27,990	0.0
Average		66,106	8.4	199,200	26.9	21,812	1.9

Considering that some of the LUBM queries return a very large number of results for large datasets, the query-time-per-result (QTPR) metric is introduced, where the query evaluation time is normalized with respect to the size of the result-set. Average QTPR represents a better overall score of the query performance on LUBM because the performance of queries with large result-sets does not dominate the score as in the case of using average query evaluation time. QTPR appearing as 0.0 indicates a value below 0.1 milliseconds.

As expected, given a large dataset, the query performance of the server is considerably better than that of the desktop. This should most probably be attributed to the better storage system and data buses between the CPUs and the main memory.

The query performance reported for AllegroGraph looks very good, especially if one considers that, based on the vendor information, “dynamic materialization” is performed during query evaluation. The major gain in terms of average QTPR and evaluation time comes from queries #2 and #9 – probably the two most challenging in the benchmark.

This is the state of the art in performance and scalability of semantic repositories. Dataset loading and query evaluation times are observed, considering the hardware configurations and the overall test environment.

7.8 Typical Applications

The applications of semantic repositories are presented in this section to illustrate their strengths across different domains and scenarios. The following table compares the most relevant characteristics of these applications.

Comparison of sample semantic repository applications

	FactForge	LinkedLifeData	BBC's World Cup website
Scope	General	Domain-specific	Domain- and application-specific
Closed-world assumption	–	+	+
Data management approach	Integration with minimal intervention	Extensive data restructuring	Well-maintained schema and data, extended using entity extraction
Update rate	Once in a few months	Once in a few months	Hundreds of updates per hour

7.8.1 Reason-Able Views to the Web of Linked Data

Reason-able views represent an approach for reasoning and the management of linked data from the Linking Open Data (LOD) cloud (see [Sect. 7.1.4](#)). *Reason-able view (RAV)* is an assembly of independent datasets, which can be used as a single body of knowledge with respect to reasoning and query evaluation.

Reason-able views are necessary as reasoning with linked data is problematic with respect to different dimensions. For example, reasoning with the Web of linked data is not feasible because of the clash between the mainstream reasoning techniques and the WWW-like nature of the data such as Linking Open Data (LOD). The major obstacles for this lay in several reasons:

- Most of the popular reasoners implement sound and complete inference under “closed-world assumption.” Such setups are irrelevant in an environment, like the Web of Data and the WWW, where the knowledge is incomplete by design and logical consistency is not guaranteed.
- In addition to that, the complexity of reasoning even with the simplest knowledge representation language based on description logics (DL – one of the most popular paradigms nowadays) OWL Lite is prohibitively high when applied to Linking Open Data (LOD). The great complexity of the algorithms for basic reasoning tasks indicates that they are unfeasible for application to large-scale knowledge bases and datasets.
- Further, the LOD cloud contains datasets that are not suitable for reasoning. Some of them are derived by the means of text-mining and include incorrect information, due

to the intrinsic limitations of the accuracy of the extraction techniques. Such inaccuracies are probably not a serious problem for human readers, but they can lead to significant noise and inconsistencies by automated reasoning.

- Finally, data publishers use OWL vocabulary with no account for its formal semantics. This results in long cycles in many category hierarchies.

Reasoning is practically unfeasible with distributed data as well. It is actually possible, but is usually far slower than reasoning with local data. The fundamental reason for that is related to the “remote join” problem. The remote join problem pertains to DBMS (Database Management Systems). The remote join is a method capable of executing a join across two DBMSs. The remote join problem is solved by creating a join view at the remote database, and a local synonym for this view, then retrieving through this local view. Distributed joins have caused performance problems throughout the history of distributed database support in general. That is why they cause problems in the context of reasoning with the Linking Open Data (LOD) cloud as well. In addition to this, this speed of access and service availability in distributed data can be a major issue regarding reasoning and manipulating the Web of Data in distributed environments.

Thus, the key ideas around the *reason-able views* approach focus on the following aspects:

- The grouping of the selected datasets and ontologies in a compound dataset, which becomes a single body of knowledge – *integrated dataset* – with respect to reasoning and query evaluation.
- Loading of the compound dataset in a single semantic repository in order to make query evaluation and reasoning practically feasible. It can be considered as an index, which caches parts of the Linking Open Data (LOD) cloud and provides access to the datasets included as Web search engines index WWW pages and facilitate their usage.
- The performance of inference with respect to tractable OWL dialects. Given all public results, only OWL Horst (see ▶ [Sect. 7.2.2](#))–like languages seem to be suitable for reasoning with data in the range of billions of statements.

Complying with all these aspects makes reasoning with the Linking Open Data (LOD) feasible. This is because a basic level of consistency of the data is being guaranteed, along with a guaranteed service availability because the compound dataset is loaded into a single semantic repository. This allows for the easier exploration and querying of unseen data and ensures a lower cost of entry.

The constitution of reason-able views obeys special selection criteria for the datasets, for example, the datasets must allow inference and deliver meaningful results under the semantics determined for the view. Further, it is necessary that the datasets are easy to define and isolate, for example, they must be clearly distinguishable from other datasets. In many cases, additional manipulations on the datasets like cleanups are required. The datasets must allow easy and cheap cleanup manipulations that can be performed on them in an automated or semiautomated fashion. Ultimately, the datasets must be more or less static, able to function in predictable way, opposite to database wrappers which

implement complex mappings to be reusable in unplanned contexts, such as Web-based applications or federated systems, where RDF is generated in answer to retrieval requests.

There are two implementations of the concept for linked data reason-able views:

- FactForge, in red on [▶ Fig. 7.10](#) (see [▶ Sect. 7.8.1.1](#))
- Linked Life Data (LLD) and PIKB Pathway and Interaction Knowledge Base, (see [▶ Sect. 7.8.1.2](#))

Although similar in spirit and based on identical principles, these two applications are very different use cases. FactForge is a collection of a vast amount of heterogeneous general purpose data, whereas LLD – PIKB is a domain-specific warehouse.

FactForge and LLD – PIKB are presented in greater detail in the following sections. While both of them can be seen as reason-able views to the Web of linked data (a notion introduced in [▶ Sect. 7.8.1](#)), sharing one and the same search, exploration, and querying facilities, they are in fact quite different in terms of the data management approach, assumptions, and users.

7.8.1.1 FactForge: The Upper-Level Knowledge Base

FactForge is a reason-able view to the Web of linked data, an assembly of some of the central LOD datasets, which have been selected and refined in order to:

- Serve as a useful index and entry point to the LOD cloud
- Present a good use case for large-scale reasoning and data integration

It includes DBPedia, Geonames, UMBEL, Freebase, WordNet, CIA World Factbook, Lingvoj datasets and Dublin Core (DC), SKOS (Simple Knowledge Organization System), RSS, FOAF schemas.

The datasets of FactForge are loaded into BigOWLIM, where forward-chaining and materialization are performed. BigOWLIM uses internally a rule language that supports R-entailment – “owl-max,” which extends OWL Horst, [60], to deliver expressiveness very similar to OWL 2 RL, [41]. The standard reasoning behavior of OWLIM is to update the deductive closure upon committing a transaction to the repository. Consistency checking is performed, applying the checking rules after adding all new statements and updating the deductive closure. Inconsistencies are reported accordingly. FactForge is loaded with OWLIM’s “partialRDFS” option enabled. This excludes rules supporting some of the features from RDFS and OWL, thus avoiding inferring and indexing three “trivial” statements for each URI in the repository.

Additionally, the loading of FactForge benefits from a specific feature of the BigTRREE engine (see [▶ Sect. 7.6.4](#)) that enables the engine to handle efficiently `owl:sameAs` statements in order to avoid their over-generation (see [▶ Sect. 7.2.1.5](#)). The results of the loading of FactForge can be summarized as follows:

- Number of inserted statements (NIS): 1.4 billion
- Number of stored statements (NSS), including the implicit ones: 2.2 billion
- Number of retrievable statements (NRS): 10 billion

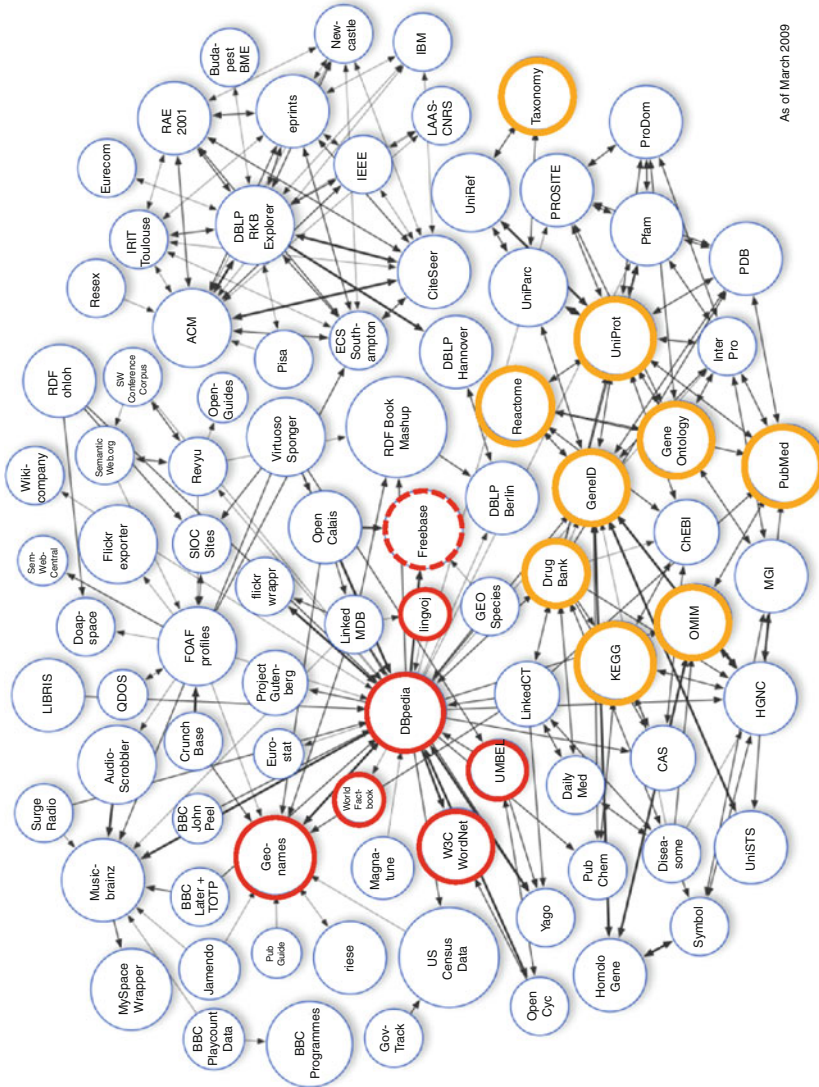


Fig. 7.10

Linking open data datasets included in reason-able views

The larger number of retrievable statements is a result of the `owl:sameAs` optimization discussed above. The optimization has “compressed” 7.8 billion statements, reducing the size of the indices five times. There are seven different “retrievable” statements against a single explicit statement asserted. The version of FactForge launched in May 2010 includes version 3.3 of DBPedia and a version of Geonames downloaded in April 2010.

FactForge is available as a free public service at <http://factforge.net>, offering the following access facilities:

- Incremental URI auto-suggest
- One-node-at-a-time exploration through Forest and Tabulator linked data browsers
- RDF Search: retrieve ranked list of URIs by keywords (see [Sect. 7.6.4](#))
- SPARQL end-point

7.8.1.2 Linkedlifedata: 25 Biomedical Databases in a Box

Linked Life Data – Pathway and Interaction Knowledge Base (LLD – PIKB, <http://linkedlifedata.com>) is the largest known reason-able view of the Web of Data (4,179,999,703 triples). It assembles a large fraction of the life science–related datasets in LOD, and includes about 20 databases, as described in [41]. Linked Life Data is a data integration platform that realizes a massive RDF warehouse solution extended with inference and semantic annotations support. It integrates semantically, molecular information and realizes its linking to the public data cloud (LOD). The well-known data sources PubMed, UMLS, Entrez-Gene, and OBO Foundry have been transformed into RDF, using, in most of the cases, SKOS schema to represent the triples [29].

LLD – PIKB contains about 2,735,148,325 explicit triples, which are complemented by another 1,444,851,378 implicit statements inferred from them.

The data integration process is linking between related resources in the disconnected datasets. It takes place automatically according to predefined alignment patterns, mapping rules. They are presented in [Sect. 7.8.1.2](#). For example, Namespace mapping identifies common parts of the URI and states that the same resource is referred to within two namespaces. Or Reference node identifies that a URI and a node with properties corresponding to parts of the URI refer to the same resource. Thus, LLD has an innovative way of handling pathways. It selects resources based on the metadata describing a resource, which interacts with the resource to be selected, whereas conventional approaches select elements based on the metadata describing them directly. Three types of mappings are supported: `exactMatch`, `closeMatch`, and `related`. `exactMatch` means that both entities have the same semantics, for example, they are both genes or proteins, and the mappings are made based on the existing stable unique identifiers. `closeMatch` means that both entities have the same semantics, for example, they are both gene or protein, but their mappings are made based on nonstable identifiers, for example, gene names. `related` means that the two mapped entities can have different semantics, for example, one of them could be a gene, and one a protein.

A fourth special way of mapping individuals is through semantic annotation. Semantic annotation is used for assigning links between the entities, recognized by arbitrary information extraction algorithms, and their semantic descriptions. This process produces metadata providing class and/or instance information about the entities. The semantic annotation links identified entities in a given LLD dataset with their mentions in documents, for example, PubMed documents, and stores this information back into the semantic repository. Furthermore, the semantic annotation in LLD is capable of identifying and resolving all alternative names of a given element or concept to the same resource. The results demonstrate efficient search capabilities over highly heterogeneous and loosely coupled domain-specific data.

LLD - PIKB is used as a domain-specific reporting tool for generating new information insights. The system facilitates the mining of concealed relations among data by interlinking information from multiple heterogeneous sources and by providing a more holistic view over a particular scientific problem.

Linked data already gained popularity as a platform for data integration and analysis in the life science and health care domain. Reference [41] reports on recent developments in the Linked Life Data (LLD) platform and the Pathway and Interaction Knowledge Base (PIKB) dataset. The main objective set for the system is to facilitate the mining of concealed relations among data. Information is mined in 15 different data sources from five different biomedical domains. More than 20 completed data sources are interconnected, thus aiding in the understanding of research problems by linking unrelated data from heterogeneous knowledge domains. The collection of domain knowledge has been optimized by the use of instance alignment patterns that restore missing information relationships in the public linked data cloud (LOD). Additionally, information from unstructured texts has been matched with semantic annotations to the linked data instances from the knowledge base. This work addresses the reality that researchers in life sciences require different views over one and the same data. To understand the “bigger picture” of a research problem, the scientists need to link visually unrelated data from heterogeneous knowledge domains, while usually the analysis is limited based on the accessible overview of the data. Semantic Web technology has a place as a promising technology for reducing the complexity of combining data from multiple sources and resolving classical integration problems related to information accessibility. RDF technologies are applied as the “semantic glue” in these processes. Linked Life Data (LLD) is a data integration platform that realizes a massive RDF warehouse solution extended with inference and semantic annotations support. It implements the RDF representation of the PubMed, UMLS, Entrez-Gene, and OBO Foundry data sources, based on the SKOS scheme, and uses linked data principles. Integration patterns have been identified to interconnect related resources in RDF database representations. Semantic Annotation has been used for assigning links between the entities, recognized by arbitrary information extraction algorithm, and their semantic descriptions. This allows knowledge acquisition based on the extraction of more complex dependencies like the analysis of relationships between entities, event, and situation descriptions. The data in Pathway and Interaction knowledge Base (PIKB) has more than 2,217 billion statements, loaded in 40 h on

a standard server configuration with an average loading and inference speed varying between 5,000 and 60,000 statements per second, depending on the complexity of the loaded dataset. Continuous updates are maintained, and all post processing activities are automated. The LLD platform demonstrates efficient search over highly heterogeneous and loosely coupled data. It is capable of executing queries that cover information from seven different sources in a timely fashion. The platform and the PIKB dataset are used as domain-specific reporting tools for generating new information insights. The Web front-end provides three paths to access the data: a Web form for issuing SPARQL queries, a browser for exploring resources, and full-text search in the graph containing the searched literal with matched resources.

7.8.2 Publishing of Content on the Web, Based on Semantic Metadata

The BigOWLIM semantic repository (see [Sect. 7.6.4](#)) was successfully integrated into the high-performance Semantic Web publishing stack powering the BBC's 2010 World Cup website. This use case is presented as an example of an application of semantic repository technology in the publishing industry, which is remarkable in two ways: It provides evidence for the advantages of the technology compared to relational databases, and it proves that such engines are already mature enough to handle high loads in critical applications.

The following information was stored in the repository backing BBC's World Cup website:

- Ontologies, both domain-specific ones (about sport and particularly football) and general (e.g., the FOAF schema)
- Factual knowledge, for example, information about specific teams, players, games, etc.
- Metadata about describing the content produced and published by the BBC by means of references to the ontologies and the entities described in the factual knowledge part

The metadata part was updated constantly to reflect at real time the stream of new content (articles and other media assets) relevant to the World Cup, which was the subject of publishing at the BBC's website. As described below, the main function of the repository was to provide selections of artifacts relevant to specific concepts – these selections were used to dynamically generate Web pages on the subject. Reasoning helped the matching between the content metadata and the subjects to take into consideration the semantics of all the data.

BigOWLIM was set up to perform materialization against a customized variant of the OWL Horst rule-set. As the updates of the repository required deletions and needed to happen in real time, the “smooth invalidation” feature of BigOWLIM (see [Sect. 7.2.1.6](#)) was critical for the performance of the overall solution. Further, the Replication Cluster feature (see [Sects. 7.4.2](#) and [Sect. 7.6.4](#)) of BigOWLIM enabled horizontal scaling with respect to the query loads and failover.

Due to confidentiality constraints, the use case will be described through citations of a couple of blog posts from the technical team at BBC that provide an insight into the

business case for the deployment of semantic technologies in their World Cup website, the technical architecture of the publishing stack, the strategic importance of the project's success, and the plans for the further usage of semantic technology and linked data within the BBC.

In [45], John O'Donovan, *Chief Technical Architect, Journalism and Knowledge, BBC Future Media & Technology*, discusses the business benefits of the implemented semantic solution:

- ▶ *"The World Cup site is our first major statement on how we think this (the Semantic Web) can work for mass market media and a showcase for the benefits it brings. . . . Though we have been using RDF and linked data on some other sites (. . .) we believe this is the first large scale, mass media site to be using concept extraction, RDF and a Triple store to deliver content."*

" . . . we are not publishing pages, but publishing content as assets which are then organized by the metadata dynamically into pages, but could be re-organized into any format we want much more easily than we could before. . . . There is also a change in editorial workflow for creating content and managing the site. This changes from publishing stories and index pages, to one where you publish content and check the suggested tags are correct. The index pages are published automatically. This process is what assures us of the highest quality output, but still saves large amounts of time in managing the site and makes it possible for us to efficiently run so many pages for the World Cup."

"As more content has Linked Data principles applied to it . . . the vision of a Semantic Web moves closer. Importantly, what we have been able to show with the World Cup, is that the technology behind this is ready to deliver large scale products."

"This is more than just a technical exercise – we have delivered real benefits back to the business as well as establishing a future model for more dynamic publishing which we think will allow us to make best use of our content and also use Linked Data to more accurately share this content and link out to other sites and content, a key goal for the BBC. We look forward to seeing the use of Linked Data grow as we move towards a more Semantic Web."

In a following post [51], Jem Rayfield, Senior Technical Architect, BBC News and Knowledge, provides more information on the technical architecture of the high-performance publishing stack and the related data flows and data modeling:

- ▶ *"The World Cup 2010 website is a significant step change in the way that content is published. . . . As you navigate through the site it becomes apparent that this is a far deeper and richer use of content than can be achieved through traditional CMS-driven publishing solutions."*

"The site features 700-plus team, group and player pages, which are powered by a high-performance dynamic semantic publishing framework. This framework facilitates the publication of automated metadata-driven web pages that are light-touch, requiring minimal journalistic management, as they automatically aggregate and render links to relevant stories."

"The foundation of these dynamic aggregations is a rich ontological domain model. The ontology describes entity existence, groups and relationships between the things/concepts that describe the World Cup. For example, "Frank Lampard" is part of the "England Squad" and the

“England Squad” competes in “Group C” of the “FIFA World Cup 2010”. The ontology also describes journalist-authored assets (stories, blogs, profiles, images, video and statistics) and enables them to be associated to concepts within the domain model . . .”

“A RDF triplestore (ref. BigOWLIM) and SPARQL approach was chosen over and above traditional relational database technologies due to the requirements for interpretation of metadata with respect to an ontological domain model. The high level goal is that the domain ontology allows for intelligent mapping of journalist assets to concepts and queries. The chosen triple store provides reasoning following the forward-chaining model and thus implied inferred statements are automatically derived from the explicitly applied journalist metadata concepts.”

“This inference capability makes both the journalist tagging and the triple store powered SPARQL queries simpler and indeed quicker than a traditional SQL approach. Dynamic aggregations based on inferred statements increase the quality and breadth of content across the site. The RDF triple approach also facilitates agile modeling, whereas traditional relational schema modeling is less flexible and also increases query complexity.”

“Our triple store is deployed multi-data center in a resilient, clustered, performant and horizontally scalable fashion, allowing future expansion for additional ontologies and indeed linked open data (LOD) sets. . . . The triple store is abstracted via a JAVA/Spring/CXF JSR 311 compliant REST service. . . . The API is designed as a generic facade onto the triple store allowing RDF data to be re-purposed and re-used pan BBC. This service orchestrates SPARQL queries and ensures that results are dynamically cached with a low ‘time-to-live’ (TTL) (1 minute) expiry cross data center using memcached.”

“This dynamic semantic publishing architecture has been serving millions of page requests a day throughout the World Cup with continually changing OWL reasoned semantic RDF data. The platform currently serves an average of a million SPARQL queries a day with a peak RDF transaction rate of 100s of player statistics per minute. . . .”

“The development of this new high-performance dynamic semantic publishing stack is a great innovation for the BBC as we are the first to use this technology on such a high-profile site. It also puts us at the cutting edge of development for the next phase of the Internet, Web 3.0.”

7.9 Related Resources

This section provides references to publications related to semantic repositories, their functionality, design, and performance.

Kiryakov [32] aims to define criteria for the validation of the performance of semantic repositories and in particular for the data layer of the LarKC platform for web-scale reasoning. It introduces the first version of the *conceptual framework for semantic repository tasks and performance aspects* (presented in [Sect. 7.3](#) here) and provides an overview of the state-of-the-art repositories. Finally, he defines *performance and scalability targets* for the development of the semantic repositories over a period of 3 years.

In [15], the developers of the Virtuoso engine (see [Sect. 7.6.8](#)) argue that web-scale RDF management requires manipulations like joining, aggregation, and the filtering of data together with inference and on-the-fly schema mapping. Further, they present some of the research and experiments on the usage of various indexing techniques and distribution schemas. The paper motivates the usage of *bitmap indices* with key compression (to improve space efficiency) and *data sampling* at query optimization time by keeping the data in memory (to address the issue of the speed of processing). The *data partitioning* schema used in Virtuoso is presented along with the basic IRI encoding and indexing mechanisms. Useful observations and statistics about the *actual computational cost* of some atomic operations and the network traffic are also presented. [16] provides more recent insights of Orri Earling on the *trends* of development of the semantic repositories and argues that in order for the latter to become a widely adopted data management technology, they should match the efficiency of the relational DBMS in dealing with regular data. As an approach to achieve such efficiency, it proposes *column-based compression* techniques and reports evaluation results of an early implementation of such technique.

As discussed in [Sect. 7.4.2](#), approaches based on distribution via data partitioning have some intrinsic limitations when it comes to full-cycle data management (see [Sect. 7.3.4](#)). Still, in specific scenarios and for specific tasks, distribution can deliver amazing results. In the scope of the LarkC project, a group at the VU Amsterdam developed the WebPIE (The distributed materialization approach implemented in WebPIE is presented in greater detail in [KR and Reasoning on the Semantic Web: Web-scale Reasoning](#)) system, which implements a MapReduce (MapReduce is a framework for the parallel and distributed processing of batch jobs on a large number of compute nodes.) based *distributed materialization*. As reported in [61], WebPIE demonstrates extremely scalable reasoning with respect to the semantics of RDFS and OWL Horst. The success of WebPIE is based on several optimizations, related to the specificity of the entailment rules, which defined the semantics of the above-mentioned languages, and assumptions about the specific data loading discipline. These optimizations allow WebPIE to decrease the number of inference jobs to be performed by the cluster, and thus the time required for closure computation. Given real-world datasets like UNIPROT and FactForge (see [Sects. 7.5.2](#) and [7.5.3](#)) of size close to 1 billion statements, it delivers OWL Horst reasoning speeds in the range of 70,000 explicit statements per second on a 64-node cluster. The experiments with the synthetic datasets of the LUBM benchmark (see [Sect. 7.5.1](#)) demonstrate speeds in the range of 500,000 statements per second for a dataset of 100 billion explicit statements. As already discussed in [Sect. 7.5.1](#), this proves that LUBM datasets are relatively easy to reason with; still, at present, this experiment demonstrates the highest speed and scalability officially reported.

WebPIE builds on top of the experience with MARVIN [48], a system, which implements *incomplete distributed materialization*. MARVIN is based on the approach of divide-conquer-swap, for example, peers autonomously partition the problem in some manner, each operate on some subproblem to find partial solutions, and then repartition their part and swap it with another peer; all peers keep repartitioning, solving, and swapping to

find all solutions. MARVIN guarantees eventual completeness of the inference process and produces its results gradually. Initial partitioning is accomplished by reading some random data from disk, and subsequent partitioning is dictated by step swap. The conquer phase is performed by an off-the-shelf reasoner which computes the closure of the portion of the data available at a specific step in a single node. The problem of making distributed reasoning scalable and load-balanced is addressed with the SpeedDate approach that is also used in newer developments around WebPIE, [35], which count on the so-called elastic clustering to deal with *skewed term frequency distributions*.

In [28] Adam Jacobs examines some facets of data organization and processing, relating to the design of the data structure for efficient querying and analysis, and not just for storing. He argues for adopting *sequential access to data*, because what makes the big data big is the repeated observations over time and space. In other words, large datasets have *inherent temporal or spatial dimensions*, and in order to achieve acceptable performance for highly order-dependent queries on truly large data, one should turn to a data representation model that recognizes the concept of inherent ordering of data down to the implementation level. Further, the paper presents a discussion on several issues related to the interplay between the constraints and the capabilities of the current hardware infrastructure and few of the most popular approaches for high-performance data management, for example, “everything in memory” and distributed computation.

The YARS repository was the first one to demonstrate an efficient implementation of *large-scale RDF management via data partitioning*. The system was proven in scale-up experiments on seven billion synthetically generated statements over 16 Index Managers on 16 machines with 100 queries with a randomly chosen resource joined with one or two quad patterns. The basic design principles behind YARS2, along with algorithms and evaluation data, are presented in [23]. While YARS is not provided at present as a stand-alone product, the results and the principles discussed in this paper are used in several of the prominent engines discussed in [▶ Sect. 7.6](#). Further, YARS was developed as part of SWSE – an end-to-end *Semantic Web search engine* that uses a graph data model to enable interactive query answering over structured and interlinked data collected from many disparate sources on the Web. In many aspects, SWSE is similar to the FactForge linked data service, presented in [▶ Sect. 7.8.1.1](#). The major difference is that the RDF data indexed in SWSE are crawled from the Web, while FactForge is loaded with specific datasets, which are preprocessed and cleaned up in order to provide some level of guarantee about the consistency of the data and inferences on top of it; it is also the case that FactForge supports SPARQL queries, while SWSE only supports keyword search. What is common between the two systems is that in both projects *ranking and information retrieval* (More information about search and retrieval methods appropriate for the retrieval of relevant information from large RDF dataset can be found in [▶ Semantic Technology Adoption: A Business Perspective: The Cases of Swoogle and Watson](#), which presents two other semantic Web search projects. Sindice (<http://sindice.com/>) is another Semantic Web search engine. The similarity between Sindice and SWSE is that they both offer partial support for structured queries: Users are allowed to make one-step attribute-value constraints, which allow efficient implementation without expensive DBMS-alike

join operations.) methods for semi-structured RDF data represent a key feature. A detailed and up-to-date description of SWSE can be found in [26], while [24] presents a brief overview of the key points and the recent advances in both SWSE and YARS.

This 2004 paper [22] presents evaluation work of knowledge-based systems in large OWL applications to help in the selection of the most appropriate system for a given task. It provides the rationale for and describes the LUBM (Lehigh University Benchmark) – today’s most popular *semantic repository benchmark*, discussed in [▶ Sect. 7.5.1](#). This is first-of-a-kind experiment with respect to the scale of data it was designed for, the open and comprehensive design, and the clear documentation of the benchmark framework, which allowed many repository vendors and users to adopt it for the testing of their tools and environments.

Over the last couple of years, semantic technologies have started getting real attention from the *business* (The adoption of semantic technologies by the industry is discussed in [▶ Semantic Technology Adoption: A Business Perspective](#)). As discussed in [▶ Sect. 7.1.1](#), RDF-based semantic repositories are seen as an alternative technology, which can replace relational DBMS in many environments in order to improve the efficiency of data integration and heterogeneous data management. RDF-based data management is being promoted by respected *mainstream analysts and consultants* such as PriceWaterhouseCoopers [49], and Gartner [63]. According to [49], among the most critical business problems today are the information gaps, especially in the areas of customer needs and business risk. What is actually missing is more context that explains what the data mean. The Semantic Web directly engages with the meaning and context of a business – its semantics, and the linked data approach ensures access to comprehensive data and a greater sharing of internal data. Data federation for web-scale many-to-many sharing with easier connections to more sources, combined with the ability to be reused by others, is the preferred data model for accomplishing this. It is argued that business data integration must be rethought as data linking, a decentralized, federated approach that uses ontology-mediated links to have those data at their sources. The goal is to create an information mediation layer that lets business staff explore what-if scenarios, assess strategies and risks, and gain insight from the messy reality of the world inside and outside the company. The linked data approach is contrasted with the data warehouse approach, which is deemed as nonflexible and outdated, unable to meet the actual business needs.

According to [63], Master Data Management (MDM) is one possible path to bridge the gap of the adoption of semantic technologies in the enterprise. It is considered as a semantic-oriented discipline focusing on sustaining a “single view” of critical enterprise information (referred as “master data”), which seems like an argument for mapping semantics across different systems and data stores. The authors believe that the link between semantics, Semantic Web, and MDM will increase in the future.

7.10 Future Issues

Semantic repositories are database management systems, based on RDF as data representations model. They combine important features of several other types of tools: reasoning

capabilities, like those of the inference engines; capabilities to handle sparse data and evolving data schemas like those of the column-stores; the robustness of the relational DBMS. After 10 years of development, semantic repositories now start seeing adoption in real-world applications, which can be explained by two reasons: The tools passed some threshold of maturity, and the market finally started understanding and appreciating their unique value proposition.

Facing real usage and actual end-user requirements from wide range of applications generates new requirements for semantic repositories. Here follows a list of directions for future development:

- Web-scale and –style incomplete reasoning, similar to those developed in the LarKC project
- Content-based retrieval modalities, like the RDF Search employed in FactForge (▶ Sect. 7.8.1.1)
- Extensible architectures, which allow for efficiently handling specific, filtering and lookup criteria, for instance, geo-spatial constraints, full-text search, and social network analysis.

7.11 Cross-References

- ▶ KR and Reasoning on the Semantic Web: RIF
- ▶ KR and Reasoning on the Semantic Web: OWL
- ▶ KR and Reasoning on the Semantic Web: Web-scale Reasoning
- ▶ Semantic Annotation and Retrieval: Web of Data
- ▶ Semantic Technology Adoption: A Business Perspective
- ▶ Semantic Web Search Engines

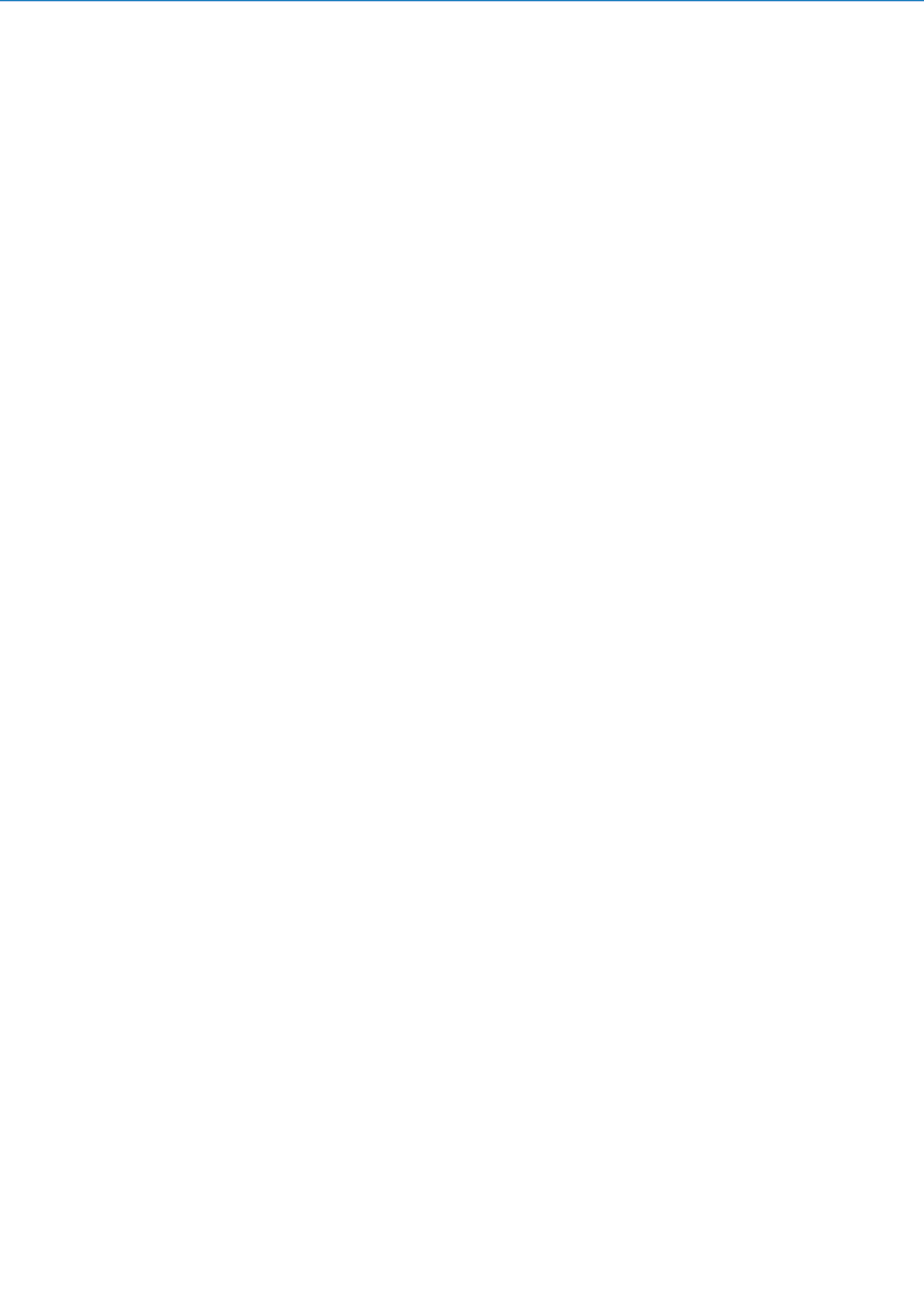
References

1. Aasman, J.: AllegroGraph 4.0 – industry’s first real time RDF store. Presentation at Semantic Technologies Conference (SemTech 2009), San Jose (2009)
2. Andersson, B., Momtchev, V.: LarKC requirements summary and data repository. LarKC project deliverable D7a.1.1. http://www.larkc.eu/wp-content/uploads/2008/01/larkc_d7a-11_requirements-summary-and-data-repository_m6.pdf (2008)
3. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Scheider, P.: The Description Logic Handbook. Theory, Implementation, Applications. Cambridge University Press, Cambridge (2003)
4. Berners-Lee, T., Fielding R., Masinter L.: Uniform Resource Identifier (URI): generic syntax. Network Working Group, Request for comments: 3986. <http://tools.ietf.org/html/rfc3986> (2005). Accessed Jan 2005
5. Berners-Lee, T.: Design issues: linked data. <http://www.w3.org/DesignIssues/LinkedData.html> (2006)
6. Bizer, Ch., Schultz, A.: Benchmarking the performance of storage systems that expose SPARQL endpoints. In: Proceedings of the Fourth International Workshop on Scalable Semantic Web knowledge Base Systems (SSWS 2008), Karlsruhe (2008)

7. Bizer, Ch., Schultz, A.: Berlin SPARQL benchmark results. Document version 1.2. <http://www4.wiwiw.fu-berlin.de/bizer/BerlinSPARQLBenchmark/results/index.html> (2009). Accessed 23 Mar 2009
8. Bizer, Ch., Schultz, A.: BSBM results for Virtuoso, Jena TDB, BigOWLIM. <http://www4.wiwiw.fu-berlin.de/bizer/BerlinSPARQLBenchmark/results/V5/index.html> (2009). Accessed 30 Nov 2009
9. Brewer, E.: Towards robust distributed systems. Keynote at Proceedings of the 19th Annual ACM Symposium on Principles of Distributed Computing (PODC 2000), Portland. <http://www.cs.berkeley.edu/~brewer/cs262b-2004/PODC-keynote.pdf> (2000)
10. Brickley, D., Guha, R.V. (eds.): Resource Description Framework (RDF) schemas. W3C Recommendation. <http://www.w3.org/TR/rdf-schema/> (2004). Accessed 10 Feb 2004
11. Carroll, J.J., Bizer, B., Hayes, P., Stickler, P.: Named graphs, provenance and trust. In: Proceedings of the 14th International conference on World Wide Web (WWW 2005), Chiba. <http://www2005.org/cdrom/docs/p613.pdf> (2005)
12. Chang, F., Dean, J., Ghemawat, S., Hsieh, W.C., Wallach, D.A., Burrows, M., Chandra, T., Fikes, A., Gruber, R.E.: Bigtable: a distributed storage system for structured data. In: Proceedings of the Seventh Symposium on Operating Systems Design and Implementation (OSDI 2006), Seattle. <http://labs.google.com/papers/bigtable.html> (2006)
13. Bechofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F., Stein, L.A.: In: Dean, M., Schreiber, G. (eds.), OWL web ontology language reference, W3C Recommendation. <http://www.w3.org/TR/owl-ref/> (Feb 2004)
14. Erling, O.: LUBM and Virtuoso. OpenLink software product blog. <http://www.openlinksw.com/dataspace/oerling/weblog/Oerri%20Erling's%20Blog/1562> (2009). Accessed 24 July 2009
15. Erling, O., Mikhailov, I.: Towards web-scale RDF. <http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VOSArticleWebScaleRDF> (2009)
16. Erling, O.: Directions and challenges for Semdata. In: Proceedings of the Semantic Data Management (SemData 2010) Workshop at the 36th International Conference on Very Large Data Bases (VLDB 2010), Singapore (2010)
17. Fischer, F., Keller, U., Kiryakov, A., Huang, Z., Momtchev, V., Simperl, E.: Initial knowledge representation formalism. LarKC project deliverable D1.1.3. http://www.larkc.eu/wp-content/uploads/2008/01/larkc_d113-initial-knowledge-representation-formalism_m7.pdf (2008)
18. Franz Inc.: AllegroGraph RDFStore 4.0. AllegroGraph product information. <http://www.franz.com/agraph/allegrograph/> (2010). Accessed 22 Aug 2010
19. Franz Inc.: AllegroGraph RDFStore version 4.0 LUBM benchmark results. http://www.franz.com/agraph/allegrograph/agraph_bench_lubm.lhtml (2010). Accessed 22 Aug 2010
20. Franz Inc.: RDFS++ dynamic materialization. <http://www.franz.com/agraph/allegrograph/dynamic-materialization.lhtml> (2010). Accessed 22 Aug 2010
21. Groszof, B.N., Horrocks, I., Volz, R., Decker, S.: Description logic programs: combining logic programs with description logic. In: Proceedings of the 12th International Conference on World Wide Web (WWW 2003), Budapest (2003)
22. Guo, Y., Pan, Z., Heflin, J.: An evaluation of knowledge base systems for large OWL datasets. *J. Web Semant.* 3(2), 158–182 (2004)
23. Harris, S., Lamb, N., Shadbolt, N.: 4store: the design and implementation of a clustered RDF store. In: Proceedings of the Fifth International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2009) at the Eighth International Semantic Web Conference (ISWC 2009), Chantilly. Lecture Notes in Computer Science, vol. 5823. Springer, Berlin (2009)
24. Harth, A., Umbrich, J., Hogan, A., Decker, S.: YARS2: a federated repository for querying graph structured data from the web. In: Proceedings of the Sixth International Semantic Web Conference (ISWC 2007), Busan. Lecture Notes in Computer Science, vol. 4825, pp. 211–224. Springer, Berlin (2007)
25. Harth, A: SWSE/YARS@SemData Sofia 2010. SemData roundtable, Sofia. <http://semdata.org/sites/default/files/harth-swse-sofia-2010.pdf> (2010). Accessed 12 Mar 2010
26. Hayes, P.: RDF semantics, W3C Recommendation. <http://www.w3.org/TR/2004/REC-rdf-mt-20040210/> (Feb 2004)
27. Hogan, A., Harth, A., Umbrich, J., Kinsella, S., Polleres, A., Decker, S.: Searching and browsing

- linked data with SWSE: the semantic web search engine. DERI technical report 2010-07-23. <http://www.deri.ie/fileadmin/documents/DERI-TR-2010-07-23.pdf> (2010)
28. Horrocks, I., Patel-Schneider, P.F., Bechhofer, S., Tsarkov, D.: OWL rules: a proposal and prototype implementation. *J. Web Semant.* 3, 23–40 (2005)
 29. Jacobs, A.: The pathologies of big data. *Commun. ACM* 52(8), 36–44 (2009)
 30. Jupp, S., Bechhofer, S., Kostkova, P., Stevens, R., Yesilada, Y.: Document navigation: ontologies or knowledge organisation systems? In: Proceedings of the Seventh International Workshop on Network Tools and Applications in Biology (NETTAB 2007), Pisa. A Semantic Web for Bioinformatics: Goals, Tools, Systems, Applications (2007)
 31. Kerrigan, M., Bradesko, L., Fortuna B.: Rapid prototype of the LarKC. LarKC project deliverable D5.2.1. http://www.larkc.eu/wp-content/uploads/2008/01/larkc_d521_rapid-prototype-of-the-larkc.pdf (2009)
 32. Kiryakov, A.: Measurable targets for scalable reasoning. LarKC project deliverable D5.5.1, formerly titled “Definition of validation goals for the prototyping phase”. http://www.larkc.eu/wp-content/uploads/2008/07/larkc_d551.pdf (2008)
 33. Kiryakov, A., Tashev, Z., Ognyanoff, D., Velkov, R., Momtchev, V., Balev, B., Peikov, I.: Validation goals and metrics for the LarKC platform. LarKC project deliverable D5.5.2. <http://www.larkc.eu/deliverables/> (2009)
 34. Kiryakov, A., Ognyanoff, D., Velkov, R., Tashev, Z., Peikov, I.: LDSR: materialized reasonable view to the web of linked data. In: Proceedings of the Third International Symposium on Rules, Applications and Interoperability (RuleML 2009), Las Vegas. Lecture Notes in Computer Science, vol. 5858. Springer, Berlin (2009)
 35. Kiryakov, A., Momtchev, V.: Triplesets: tagging and grouping in RDF datasets. W3C Workshop “RDF Next Steps”, Stanford (June 2010)
 36. Kiryakov, A., Bishop, B., Ognyanoff, D., Peikov, D., Tashev, Z., Velkov, R.: The features of BigOWLIM that enabled the BBC’s World Cup website. In: Proceedings of the Semantic Data Management (SemData 2010) Workshop at the 36th International Conference on Very Large Data Bases (VLDB 2010), Singapore (2010)
 37. Kotoulas, S., Oren, E., Van Harmelen, F.: Mind the data skew: distributed inferencing by speeddating in elastic regions. In: Proceedings of the 19th International Conference on World Wide Web (WWW 2010), Raleigh (2010)
 38. Heath, T.: Linked data – connect distributed data on the web. <http://linkeddata.org/> (2007)
 39. World Wide Web Consortium (W3C): Linking open data. W3C SWEO community project. <http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData/> (2007)
 40. Ma, L., Yang, Y., Qiu, Z., Xie, G., Pan, Y.: Towards a complete OWL ontology benchmark. In: Proceedings of the Third European Semantic Web Conference (ESWC 2006), Budva. Lecture Notes in Computer Science, vol. 4011, pp. 125–139. Springer, Berlin (2006)
 41. Manola, F., Miller, E. (eds.): RDF primer, W3C Recommendation. <http://www.w3.org/TR/rdf-primer/> (Feb 2004)
 42. McGuinness, D.L., van Harmelen, F. (eds.): OWL web ontology language. Overview. W3C Recommendation. <http://www.w3.org/TR/owl-features/> (2004)
 43. Momtchev, V., Kiryakov, A.: Second generation ontology representation and data integration (ORDI) framework, specification. Ontotext technical documentation. http://www.ontotext.com/ordi/ORDI_SG/ORDI_SG_Specification.pdf (2006). Accessed 13 Oct 2006
 44. Momtchev, V., Peychev, D., Primov, T., Georgiev, G.: Expanding the pathway and interaction knowledge in linked life data. In: Proceedings of the International Semantic Web Challenge (ISWC 2009), Chantilly. Lecture Notes in Computer Science, vol. 5823. Springer, Berlin. <http://challenge.semanticweb.org/> (2009)
 45. Motik, B., Cuenca Grau, B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C. (eds.): OWL 2 web ontology language profiles, W3C Candidate Recommendation. <http://www.w3.org/TR/owl2-profiles/> (Feb 2009)
 46. Motik, B., Sattler, U., Studer, R.: Query answering for OWL-DL with rules. *J. Web Semant.* 3(1), 41–60 (2005)
 47. North, K.: The NoSQL alternative. Dr. Dobb’s J. <http://www.drdoobs.com/database/224900500>. Accessed 21 May 2010

48. O'Donovan, J.: The World Cup and a call to action around linked data. BBC blog post. http://www.bbc.co.uk/blogs/bbcinternet/2010/07/the_world_cup_and_a_call_to_ac.html. Accessed 9 July 2010
49. ONTOCORE: Ontology logic and reasoning at semantic Karlsruhe. Home page, <http://logic.aifb.uni-karlsruhe.de/>
50. Ontotext Lab: RDF(S), rules and OWL dialects. http://www.ontotext.com/inference/rdfs_rules_owl.html. Accessed Dec 2009 (2009)
51. Oren, E., Kotoulas, S., Anadiotis, G., Siebes, R., Ten Teije, A., Van Harmelen, F.: MARVIN: distributed reasoning over large-scale semantic web data. *J. Web Semant.* 7(4), 305–316 (2009)
52. PricewaterhouseCoopers: Spring of the data web. *Technology Forecast. A Quart. J. Spring* 2009. <http://www.pwc.com/techforecast/> (2009)
53. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF, W3C Recommendation. <http://www.w3.org/TR/rdf-sparql-query/> (Jan 2008)
54. Rayfield, J.: BBC World Cup 2010 dynamic semantic publishing, BBC blog post. http://www.bbc.co.uk/blogs/bbcinternet/2010/07/bbc_world_cup_2010_dynamic_sem.html (2010)
55. Reed, D.P.: Naming and synchronization in a decentralized computer system. MIT dissertation, <http://portal.acm.org/citation.cfm?id=889815> (1978)
56. Ruhloff, K., Dean, M., Emmons, I., Ryder, D., Sumner, J.: An evaluation of triple-store technologies for large data stores. In: *Proceedings of the Scalable Semantic Systems Workshop (SSSW 2007)*, Portugal (2007)
57. Salvadores, M., Correndo, G., Omitola, T., Gibbins, N., Harris, S., Shadbolt, N.: 4s-reasoner: RDFS backward chained reasoning support in 4store. In: *Proceedings of the 2010 International Workshop on Web-Scale Knowledge Representation, Retrieval, and Reasoning (Web-KR3 2010)*, Toronto (2010)
58. Stoilos G., Grau B.C., Horrocks I.: How incomplete is your semantic web reasoner? In: *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI 10)*, Atlanta (2010)
59. Swiss Institute of Bioinformatics. UniProt RDF, <http://dev.isb-sib.ch/projects/uniprot-rdf/> (2009)
60. SYSTAP LLC: Bigdata: approaching web scale for the semantic web. http://www.bigdata.com/whitepapers/bigdata_whitepaper_07-08-2009.pdf (2009)
61. SYSTAP LLC: Bigdata overview. SemData roundtable, Sofia. <http://semdata.org/sites/default/files/bigdata-sofia-roundtable-3-10-2010.pdf> (2010). Accessed 12 Mar 2010
62. ter Horst, H. J.: Combining RDF and part of OWL with rules: semantics, decidability, complexity. In: *Proceedings of the Fourth International Semantic Web Conference (ISWC 2005)*, Galway. *Lecture Notes in Computer Science*, vol. 3729, pp. 668–684. Springer, Heidelberg (2005)
63. Thakker, D., Osman, T., Gohil, S., Lakin, P.: A pragmatic approach to semantic repositories benchmarking. In: *Proceedings of the Seventh Extended Semantic Web Conference (ESWC 2010)*, Heraklion (2010)
64. Todorova, P., Kiryakov, A., Ognyanoff, D., Peikov, I., Velkov, R., Tashev, Z.: Spreading activation components. LarkC project deliverable D2.4.1 (2009)
65. Urbani, J., Kotoulas, S., Maassen, J., Van Harmelen, F., Bal, H.: OWL reasoning with WebPIE: calculating the closure of 100 billion triples. In: *Proceedings of the Seventh Extended Semantic Web Conference (ESWC 2010)*, Heraklion. *Lecture Notes in Computer Science*, vol. 6088, pp. 213–227. Springer, Berlin (2010)
66. Vertica: The vertica database datasheet. Product overview. http://www.vertica.com/_pdf/VerticaDatabaseDataSheet.pdf (2010). Accessed Jan 2010
67. White, A. (Gartner): Semantic web moving ever close to the 'Semantic Enterprise?' http://blogs.gartner.com/andrew_white/2009/04/30/semantic-web-moving-ever-close-to-the-semantic-enterprise/ (2009)
68. World Wide Web Consortium (W3C): Linking open data. W3C SWEO community project home page. <http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData> (2010). Accessed Jan 2010
69. Wu, A., Wu, Z., Kolovski, V.: An enterprise inference engine inside Oracle Database 11g Release 2. Presentation at *Semantic Technology Conference (SemTech 2010)*, San Francisco (2010)



8 Querying the Semantic Web: SPARQL

Emanuele Della Valle · Stefano Ceri
Politecnico di Milano, Milano, Italy

8.1	<i>Scientific and Technical Overview</i>	301
8.1.1	Example RDF Graphs	303
8.1.2	Anatomy of a SPARQL Query	307
8.1.3	Basic Patterns	308
8.1.4	Writing a Simple Query	309
8.1.5	Matching RDF Literals	310
8.1.6	RDF Term Constraints	310
8.1.7	More Sophisticated Graph Patterns	314
8.1.8	Dealing with Blank Nodes	316
8.1.9	Negation in SPARQL	317
8.1.10	The FROM, FROM NAMED, and GRAPH Clauses	319
8.1.11	Solution Modifiers	322
8.1.12	Query Forms CONSTRUCT, ASK, and DESCRIBE	323
8.1.12.1	CONSTRUCT Form	323
8.1.12.2	ASK Form	325
8.1.12.3	DESCRIBE Form	325
8.1.13	SPARQL Protocol	326
8.1.14	SPARQL Semantics	327
8.2	<i>Example Applications</i>	329
8.2.1	Early Days	330
8.2.1.1	DBLP Bibliography Database	330
8.2.1.2	RKB Explorer	331
8.2.1.3	Semantic Web Dog Food Corpus	332
8.2.2	First Uptakers	333
8.2.3	Larger Audience Applications: DBTune Case Study	338
8.2.3.1	MusicBrainz	340
8.2.3.2	Jamendo	340
8.2.3.3	BBC Playcount Data	341
8.2.4	Addressing “The Modigliani Test”	342
8.2.5	Future Applications	342

8.3	<i>Related Resources</i>	344
8.4	<i>Future Issues</i>	346
8.4.1	Extensions to the Query Language	346
8.4.1.1	Aggregates	346
8.4.1.2	Subqueries	348
8.4.1.3	Project Expressions	349
8.4.1.4	Negation	350
8.4.1.5	Other Extensions Under Discussion	351
8.4.2	Insert, Update, and Delete	351
8.4.3	Beyond the SPARQL 1.0 Protocol	352
8.4.3.1	SPARQL Protocol	353
8.4.3.2	Uniform HTTP Protocol for Managing RDF Graphs	353
8.4.4	SPARQL Service Descriptions	354
8.4.5	Entailment Regimes	354
8.4.6	Querying the Entire Semantic Web with SPARQL	356
8.4.7	Continuous SPARQL	358
8.5	<i>Conclusions</i>	359

Abstract: SPARQL – Simple Protocol And RDF Query Language – is the language, proposed by W3C, for querying RDF data published on the Web, both stored natively or viewed via middleware. SPARQL offers a syntactically SQL-like language for querying RDF graphs via pattern matching, as well as a simple communication protocol that can be used by clients for issuing SPARQL queries against *endpoints*.

The first section provides the reader with a scientific and technical overview of the SPARQL query language. Basic concepts, such as the notions of triple and graph patterns are presented first. The section, then, shows how to write simple queries, and progressively introduces the reader to the full expressive power of SPARQL.

The second section illustrates some examples of applications, progressing in a quasi-chronological order. It starts with “early days” applications, when RDF data were lacking and the Semantic Web practitioners applied semantic technologies to bibliographic and conference data. Next, it moves on to “first uptakes” in the area of bioscience, which can be considered as the earlier science adopting Semantic Web technologies. This section is concluded by the presentation of some large applications, showing SPARQL queries that nowadays can be issued against interlinked RDF repositories about music and about governmental data.

The third section is dedicated to SPARQL implementations, in particular to those ones that are mostly used and widely deployed. It also discusses the standard compliance of the implementations, based upon the W3C test suite.

Finally, the fourth section discusses some of the issues that characterize the current development of SPARQL. It presents foreseen extensions to the query language, in particular a proposal for remotely updating RDF graphs, four vocabularies for describing SPARQL endpoints, the behavior of SPARQL under different entailment regimes, three approaches for querying the entire Semantic Web with SPARQL, and three proposals for extending SPARQL to the management of streams of rapidly flowing information.

8.1 Scientific and Technical Overview

As Richard MacManus wrote in a post [36] on ReadWriteWeb, “The tipping point for the long-awaited Semantic Web may be when you can query a set of data about someone not too famous (e.g., Modigliani), and get a long list of structured results in return”.

Amedeo Modigliani (<http://www.modigliani-foundation.org/>) is a celebrated painter of the early twentieth century but he is not as famous as Da Vinci or Picasso. Richard MacManus has challenged the Semantic Web to provide structured answers to the query: “tell me the locations of all the original paintings of Modigliani” (the so-called “Modigliani Test”): a tool retrieving such locations would be a suitable query language for the Web.

SPARQL [45] is a query language designed for the Semantic Web and is the language recommended by W3C for this use.

SPARQL is completely integrated into the Semantic Web and uses other W3C recommendations. It assumes that data are represented as RDF graphs [47], that resources are identified by IRIs (Section 3.1 of [49]), and that RDF literals are typed with XML Schema

datatype [82]. Moreover, SPARQL imports a subset of the XPath operators [81], which enable a user to test the values of RDF literals.

SPARQL offers a syntactically SQL-like language for querying RDF graphs via pattern matching, but it is far more powerful than SQL, since it is designed for the *open*, *decentralized*, and *fluid* Web.

Given that RDF data describing the same real-world objects (e.g., Modigliani's paintings) can be published in multiple websites, SPARQL specifications include not only a query language, but also a simple communication protocol [13] that can be used by a client for issuing SPARQL queries against some remote endpoint, producing answers either in RDF or in XML [5].

Moreover, given that data can be published on the Web using different vocabularies, SPARQL specifications propose four different query forms: `SELECT`, `CONSTRUCT`, `DESCRIBE`, and `ASK`.

- The `SELECT` and `CONSTRUCT` forms are suitable for issuing queries against known endpoints that expose data using known vocabularies. The `SELECT` form returns results in a tabular format using XML, whereas the `CONSTRUCT` form returns results in RDF.
- When clients do not know a resource's vocabulary but they know the IRI behind a SPARQL endpoint, they can still issue SPARQL queries using the `DESCRIBE` form. A `DESCRIBE` query returns an RDF graph describing the requested resource.
- When clients do not know which SPARQL endpoint could answer a query, they can discover it by using the `ASK` form. An `ASK` query returns “yes” if the endpoint is able to give at least an answer to the query and “no” otherwise.

The `CONSTRUCT` form can also be used to transform one vocabulary into another one, thus offering a simple solution to handling the heterogeneous vocabularies that characterize an open environment like the Web.

Finally, SPARQL can exploit some Semantic Web inference mechanisms. SPARQL 1.1, currently under specification [59], supports queries whose answers are not directly specified in the RDF graph, but that can be inferred using a set of inference rules [41, 51].

The rest of this section is organized as follows. ▶ [Section 8.1.1](#) introduces the RDF graphs that are used by the various examples of SPARQL queries. ▶ [Section 8.1.2](#) briefly introduces the overall structure (anatomy) of an SQL-like SPARQL query. ▶ [Section 8.1.3](#) presents the notions of triple and graph patterns. ▶ [Section 8.1.4](#) is dedicated to writing simple queries, while ▶ [Sects. 8.1.5](#), ▶ [8.1.6](#), ▶ [8.1.7](#), ▶ [8.1.8](#), and ▶ [8.1.9](#) progressively introduce the reader to the full expressive power of SPARQL. ▶ [Section 8.1.10](#) describes the concept of RDF dataset, that is, the RDF data against which the query is executed, and shows how RDF datasets can be referenced from a SPARQL query. ▶ [Section 8.1.11](#) presents the concept of solution modifiers, which allow one to manipulate a result, for example, ordering it. ▶ [Section 8.1.12](#) presents the `CONSTRUCT`, `ASK`, and `DESCRIBE` forms of query. ▶ [Section 8.1.13](#) briefly presents the SPARQL protocol. Finally, ▶ [Sect. 8.1.14](#) is dedicated to the semantics of SPARQL. A discussion about the “Modigliani test” is postponed to ▶ [Sect. 8.2.4](#).

8.1.1 Example RDF Graphs

The RDF data model expresses information as *graphs* consisting of *triples* with subject, predicate, and object. Before presenting SPARQL, this section introduces the examples of RDF graphs that are used in the rest of [Sect. 8.1](#). All the examples are taken from DBpedia [10]; at the time of the writing of this chapter, all SPARQL queries included in the chapter were successfully tested against the RDF graphs, which are described next.

DBpedia is a community effort for extracting structured information from Wikipedia and making this information available on the Web. The DBpedia dataset currently consists of around 274 millions of RDF triples, which have been extracted from the main national versions of Wikipedia. It describes more than two millions of “things,” including hundred thousands of “people” and “places,” and ten thousands of “music albums,” “films,” and “companies.” It features labels and short abstracts for these things in 14 different languages, over 600,000 links to images, and over 3 million links to external Web pages, and almost 5 million external links into other RDF datasets. All these things are organized according to almost 415,000 Wikipedia categories.

[Table 8.1](#) presents a subset of the RDF graph that describes the “visitor attractions of Milan” by using Turtle syntax [75], [Fig. 8.1](#) gives a visual representation of the same knowledge fragment.

Turtle syntax offers several shortcut notations. URIs can be shortened using the prefix notation (i.e., lines 1–5 are prefix declarations for the vocabularies used in the rest of the graph). Repeated subjects described with different properties can be divided from their predicates and objects by using a semicolon (i.e., the character “;” at the end of lines 7, 8, and 9). Similarly, repeated pairs of subject and predicate can be divided from their objects by using a comma (i.e., the character “,” at the end of line 10 and 11).

Lines 6–12 describe the Wikipedia category of the “visitor attractions of Milan.” Line 7 asserts that the resource is a concept in the Simple Knowledge Organization System (SKOS) Vocabulary [58]. SKOS is a light ontological language to model thesauri, classification schemes, subject heading systems, and taxonomies within the framework of the Semantic Web. The SKOS vocabulary includes properties, such as `skos:broader`, to state hierarchical relationships between concepts. This property is used at line 10, 11, and 12 to state that the concept of “visitor attractions of Milan” has as broader concepts “Milan,” “visitor attractions in Italy,” and “visitor attractions by city.” In a similar way, at line 13–16, the concepts of museums, churches, piazzas, and gardens in Milan have all “visitor attractions of Milan” as broader concepts. The property `skos:subject` is used at lines 17–21 to assert that “Biblioteca Ambrosiana,” “Brera Academy,” “Via Montenapoleone,” “Castello Sforzesco,” and “Pinacoteca di Brera” belong to the “visitor attraction in Milan” category. To complete the description, at line 8 and 9, the `skos:prefLabel` property and the `rdfs:label` property from the RDF Schema Vocabulary (another ontological language for the Semantic Web) are used to provide a human-readable label in English (see “@en” language tag [50] after the literal values).

Table 8.1

Part of the RDF graph that described the category “Visitor attractions in Milan” in DBpedia

```

1. @prefix rdf:      <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
2. @prefix rdfs:    <http://www.w3.org/2000/01/rdf-schema#>.
3. @prefix skos:    <http://www.w3.org/2004/02/skos/core#>.
4. @prefix dbpedia: <http://dbpedia.org/resource/>.
5. @prefix dbpcat:  <http://dbpedia.org/resource/Category:>.
6. dbpcat:Visitor_attractions_in_Milan
7.   rdf:type      skos:Concept ;
8.   rdfs:label   "Visitor attractions in Milan"@en ;
9.   skos:prefLabel "Visitor attractions in Milan"@en ;
10.  skos:broader dbpcat:Milan ,
11.     dbpcat:Visitor_attractions_in_Italy ,
12.     dbpcat:Visitor_attractions_by_city .
13. dbpcat:Museums_in_Milan      skos:broader dbpcat:Visitor_attractions_in_Milan .
14. dbpcat:Churches_in_Milan    skos:broader dbpcat:Visitor_attractions_in_Milan .
15. dbpcat:Piazzas_in_Milan     skos:broader dbpcat:Visitor_attractions_in_Milan .
16. dbpcat:Gardens_in_Milan     skos:broader dbpcat:Visitor_attractions_in_Milan .
17. dbpedia:Biblioteca_Ambrosiana skos:subject dbpcat:Visitor_attractions_in_Milan .
18. dbpedia:Brera_Academ        skos:subject dbpcat:Visitor_attractions_in_Milan .
19. dbpedia:Via_Montenapoleone  skos:subject dbpcat:Visitor_attractions_in_Milan .
20. dbpedia:Castello_Sforzesco  skos:subject dbpcat:Visitor_attractions_in_Milan .
21. dbpedia:Pinacoteca_di_Brer   skos:subject dbpcat:Visitor_attractions_in_Milan .

```

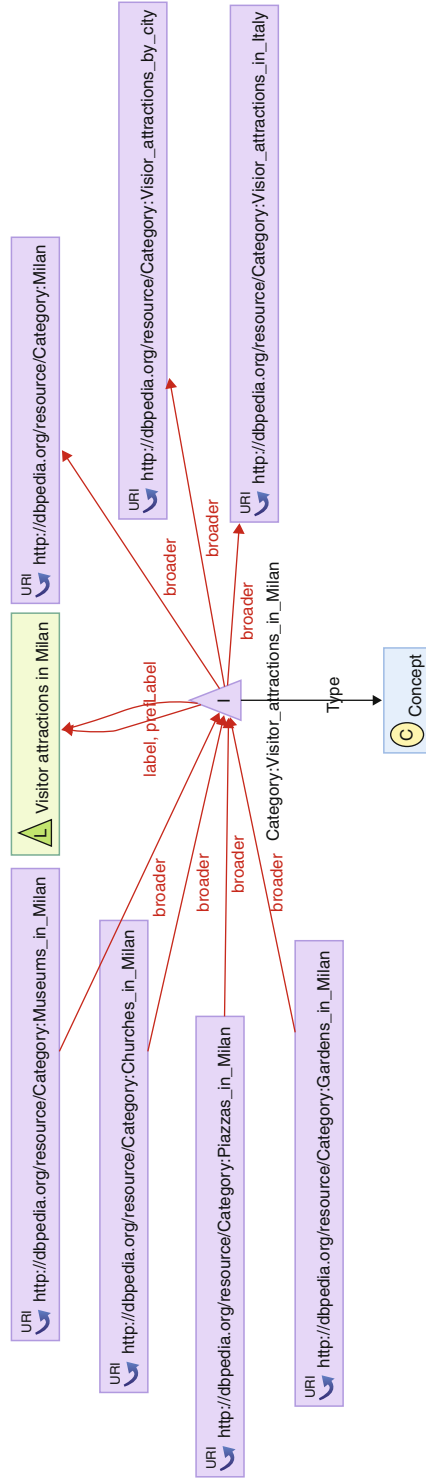


Fig. 8.1

A graphical representation of the RDF graph describing "Visitor attractions in Milan" from DBpedia

Such redundancy is quite normal in the Semantic Web because it increases the possibility to retrieve an answer when issuing a query. The complete file described above is available online at: http://dbpedia.org/data/Category:Visitor_attractions_in_Milan.rdf.

The RDF graph in ▶ [Table 8.2](#) contains a short description of the Milan Cathedral. Lines 7 and 8 assert that Milan Cathedral belongs to the category of the “churches in Milan.” Line 9 asserts the same notion, but it does so referring the YAGO knowledge base [71], a huge semantic knowledge base automatically generated from WorldNet and Wikipedia that has a manually confirmed accuracy of 95%. At lines 10 and 11, the resource is described with two labels, one in Italian (see “@it” at line 10) and one in English (see “@en” at line 11). Line 12 asserts, using the DBpedia-specific property `capacity`, that the Milan Cathedral can host 40,000 people. Note that the literal is typed using a datatype IRI specified in XML Schema [82]. The complete file described above is available online at: http://dbpedia.org/data/Milan_Cathedral.rdf.

The RDF graph in ▶ [Table 8.3](#) contains a short description of another monumental church in Milan named “Basilica di Sant’Ambrogio.” In this RDF graph, a W3C

■ **Table 8.2**

Part of the RDF graph that described the “Milan Cathedral” in DBpedia

```

1. @prefix dbpprop: <http://dbpedia.org/property/> .
2. @prefix dbpedia: <http://dbpedia.org/resource/> .
3. @prefix dbcatt: <http://dbpedia.org/resource/Category:> .
4. @prefix yago: <http://dbpedia.org/class/yago/> .
5. @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
6. @prefix skos: <http://www.w3.org/2004/02/skos/core#> .
7. dbpedia:Milan_Cathedral
8.   skos:subject dbcatt:Churches_in_Milan ;
9.   skos:subject yago:ChurchesInMilan ;
10.  rdfs:label "Duomo di Milano"@it ;
11.  rdfs:label "Milan Cathedral"@en ;
12.  dbpprop:capacity "40000"^^xsd:integer

```

■ **Table 8.3**

Part of the RDF graph that described the “Basilica of Sant’Ambrogio” in DBpedia

```

1. @prefix dbpedia: <http://dbpedia.org/resource/> .
2. @prefix dbcatt: <http://dbpedia.org/resource/Category:> .
3. @prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#> .
4. @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
5. @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
6. dbpedia:Basilica_of_Sant%27Ambrogio
7.  rdf:type dbcatt:Churches_in_Milan ;
8.  rdfs:label "Basilica of Sant'Ambrogio"@en ;
9.  geo:long "9.1758"^^xsd:float ;
10. geo:lat "45.4624"^^xsd:float .

```

vocabulary for geo-positioning [27] was used to represent latitude and longitude of the church using WGS84 as a reference datum. The complete file is available online at http://dbpedia.org/data/Basilica_of_Sant%27Ambrogio.rdf.

8.1.2 Anatomy of a SPARQL Query

A SPARQL query is composed of five parts (see [Fig. 8.2](#)): zero or more prefix declarations, a query result clause, zero or more FROM or FROM NAMED clauses, a WHERE clause, and zero or more query modifiers.

The optional PREFIX declarations introduce shortcuts for long IRIs as normally done when working with XML namespaces. Such prefixes can be used in the WHERE clause.

The query result clause specifies the form of the results. As detailed in [Sect. 8.1.9](#), a SPARQL query can take four forms: SELECT, ASK, CONSTRUCT, and DESCRIBE. SELECT queries provide answers in a tabular form as if the SPARQL query were a SQL query executed against a relational database. The ASK form checks whether the SPARQL endpoint can provide at least one result; if it does, the answer to the query is YES, otherwise the answer is NO. The CONSTRUCT form is similar to the SELECT form, but it provides the answer to the query as an RDF graph. The DESCRIBE form is conceived to retrieve information from a SPARQL endpoint without knowing the vocabulary in use, producing as result an RDF graph. The optional set of FROM or FROM NAMED clauses define the dataset against which the query is executed.

The WHERE clause is the core of a SPARQL query. It is specified in terms of a set of triple patterns. As extensively explained in the following sections, these triple patterns are used to select the triples composing the result.

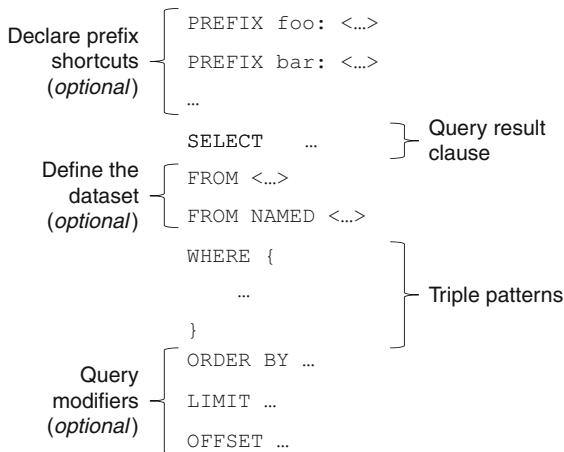


Fig. 8.2

The anatomy of a SPARQL query

Finally, the set of optional query modifiers operate over the triples selected by the `WHERE` clause, before generating the result. As in SQL, the clause `ORDER BY` orders the results set, the `LIMIT` and `OFFSET` allow getting results in chunks.

8.1.3 Basic Patterns

SPARQL is a query language for *pattern matching* against RDF graphs. The core of most forms of SPARQL query contains at least a *triple pattern*. A triple pattern is like an RDF triple except that the subject, the predicate, and the object may be a variable. Syntactically a variable is formed by a question mark “?” followed by a term, for example, `?name`. A triple pattern matches the triples in the RDF data whose RDF terms may be substituted for the variables. For instance, the triple pattern (1), when issued against the graph in [Table 8.1](#), matches the triples at line 13, 14, 15, and 16; the triple pattern (2) matches the previous four triples as well as the next five ones (i.e., from line 13 to line 21).

[Table 8.1](#) matches the triples at line 13, 14, 15, and 16; the triple pattern (2) matches the previous four triples as well as the next five ones (i.e., from line 13 to line 21).

```
(1) ?s skos:broader dbpedia:Visitor_attractions_in_Milan .
(2) ?s ?p dbpedia:Visitor_attractions_in_Milan .
```

A set of triple patterns together form a *basic graph pattern*. A basic graph pattern matches a subgraph of the RDF data when the variables in the graph pattern substitute the RDF terms. For instance, the basic graph pattern (3) matches subgraphs in which the same resource is described to be of type `skos:concept` (at line 1) and to have at least one broader concept (at line 2).

```
(3) 1. ?s rdf:type skos:Concept .
    2. ?s skos:broader ?o .
```

The basic graph pattern (3), when issued against the graph in [Table 8.1](#) matches the subgraph represented in [Table 8.4](#). Specifically, the triple pattern at line 1 matches only one time and results in the triple at line 1 in [Table 8.4](#), whereas the triple pattern at line 2 matches three times and results in the triples at line 2, 3, and 4 in [Table 8.4](#).

Table 8.4

Triples in the graph of [Table 8.1](#) matching with the basic graph pattern (3)

```
1. dbpcat:Visitor_attractions_in_Milan rdf:type skos:Concept ;
2.   skos:broader dbpcat:Milan ;
3.   skos:broader dbpcat:Visitor_attractions_in_Italy ;
4.   skos:broader dbpcat:Visitor_attractions_by_city .
```

8.1.4 Writing a Simple Query

The basic graph pattern (3) can be used to write a simple SPARQL query that selects from the graph in [▶ Table 8.1](#) the DBpedia category representing “visiting attraction in Milan” as well as the three broader DBpedia categories representing respectively “Milan,” “visitor attractions in Italy” and “visitor attractions by city.”

At lines 1 and 2, the keyword `PREFIX` is used to associate the prefixes `skos` and `rdf` with the respective IRI. The syntax is similar, but not identical (in SPARQL, different from Turtle syntax, the `PREFIX` keyword is not preceded by the “@” character and the declaration is not closed by the “.” character) to the Turtle RDF syntax used in [▶ Table 8.1](#). The declared prefixes can be used in the query when writing RDF terms. For those prefixes which are extensively used, the `BASE` keyword can be used to define a default namespace which will apply to all RDF terms without prefix.

At line 3, the `SELECT` clause is used to define the output of the query. When the `SELECT` form for SPARQL queries is used, the result is a set of tuples that must appear in the graph patterns and match the variables of the `SELECT` clause (in the example, the variables `?s` and `?o`, separated by a blank space). Note that not all the variables used in expressing the graph patterns must be present in the `SELECT` clause; if a variable is present in the `SELECT` clause but unused in the graph pattern, it is left unbound in the result.

At line 4, the `FROM` clause is used to indicate to the query processor the location of the RDF data that should be queried (in the example, an external RDF graph that is available at the URL indicated between angular brackets).

Finally, at line 5, the `WHERE` clause is used to specify the graph patterns, that is, the part of the query used to match a set of subgraphs in the RDF data. More sophisticated ways to express graph patterns will be presented in [▶ Sect. 8.1.7](#).

By issuing the SPARQL query in [▶ Table 8.5](#) against the RDF data in [▶ Table 8.1](#), three-rows table showed in [▶ Table 8.6](#), corresponding to three bindings for variables `?s` and `?o`, is obtained; note that the bindings are extracted from triples in [▶ Table 8.4](#), that – as discussed earlier – match with the `WHERE` clause of the query.

■ **Table 8.5**

Basic SPARQL query that uses the basic graph pattern (3)

```

1. PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
2. PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
3. SELECT ?s ?o
4. FROM <http://dbpedia.org/data/Category:
   Visitor_attractions_in_Milan.rdf>
5. WHERE {?s rdf:type skos:Concept.
6.        ?s skos:broader ?o . }
```

■ **Table 8.6**

Variable bindings obtained by issuing the query in Table 4 against the RDF data in Table 1

?s	?o
<code>dbcat:Visitor_attractions_in_Milan</code>	<code>dbcat:Visitor_attractions_by_city</code>
<code>dbcat:Visitor_attractions_in_Milan</code>	<code>dbcat:Visitor_attractions_in_Italy</code>
<code>dbcat:Visitor_attractions_in_Milan</code>	<code>dbcat:Milan</code>

8.1.5 Matching RDF Literals

Matching literals in SPARQL (and more in general in RDF, in the context of typing and namespaces) is quite intricate; rather than giving lengthy definitions, this section gives the reader an intuition through some examples. The data in [Table 8.2](#), which describe Milan Cathedral, contain three RDF literals: “Milan Cathedral”@en is a label in English; “Duomo di Milano”@it is a label in Italian; and “40000”^^xsd:integer is an integer that represents the number of people that can be accommodated in the cathedral (or simply the “capacity” of the cathedral).

The SPARQL query in [Table 8.7a](#) has no results because “Milan Cathedral” is not the same RDF literal as “Milan Cathedral”@en. On the contrary, when issuing the SPARQL query in [Table 8.7b](#), in which the language tag @en is explicitly expressed, the variable ?s gets bound to `dbpedia:Milan_Cathedral`.

In general, in order to match an arbitrary datatype, it is necessary to write such a datatype explicitly in the query using the ^^ notation. For instance, the query in [Table 8.8](#), selects those resources whose capacity is described using the integer 40000. The query processor has no need to understand the values in the space of the datatype: if both the lexical form (before ^^) and the IRI (after ^^) of an RDF literal in the pattern are identical, then such RDF literal matches the RDF term in the data.

In the case of integer values, the datatype can be omitted, because 40000 is a shortened form for “40000”^^xsd:integer. Thus, the query in [Table 8.9](#) is equivalent to the one in [Table 8.8](#).

8.1.6 RDF Term Constraints

SPARQL allows restrictions of the solutions by constructing constraints within `FILTER` clauses. An RDF term bound to a variable appears in the results if the constraint in the `FILTER` expression, applied to the term, evaluates to `TRUE`.

SPARQL grammar supports the construction of constraints in different ways. SPARQL imports a subset of the XPath operators [81], which enable a user to write constraints on literals typed with XML Schema [82] datatype IRIs, such as `xsd:string`, `xsd:integer`, `xsd:decimal`, `xsd:float`, and `xsd:dateTime`. SPARQL grammar also includes operators that support the additional datatypes imposed by the RDF data model (i.e., IRI, literal,

■ **Table 8.7**

Two SPARQL queries, the first gives no results because the RDF literal “Milan Cathedral” does not include the language tag @en

a)	<pre>1. PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> 2. SELECT ?s 3. FROM <http://dbpedia.org/data/Milan_Cathedral.rdf> 4. WHERE { ?s rdfs:label "Milan Cathedral" . }</pre>
b)	<pre>1. PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> 2. SELECT ?s 3. FROM <http://dbpedia.org/data/Milan_Cathedral.rdf> 4. WHERE { ?s rdfs:label "Milan Cathedral"@en . }</pre>

■ **Table 8.8**

SPARQL query that matches resources whose capacity is the integer 40000

<pre>1. PREFIX dbpprop: <http://dbpedia.org/property/> 2. SELECT ?s 3. FROM <http://dbpedia.org/data/Milan_Cathedral.rdf> 4. WHERE { ?s dbpprop:capacity "40000"^^xsd:integer }</pre>

■ **Table 8.9**

SPARQL query that uses 40000 as a shortened form for “40000”^^xsd:integer

<pre>1. PREFIX dbpprop: <http://dbpedia.org/property/> 2. SELECT ?s 3. FROM <http://dbpedia.org/data/Milan_Cathedral.rdf> 4. WHERE { ?s dbpprop:capacity 40000 . }</pre>
--

blank nodes, and variables). A comprehensive list of these operators is presented in [▶ Table 8.11](#) together with the operand datatype.

As a simple example of filtering based on XPath operators consider the query in [▶ Table 8.10](#) that selects the cathedrals whose capacity is above 50,000. As in the query in [▶ Table 8.8](#), the variable `?s` is bound to `dbpedia:Milan_Cathedral` and, thus, the variable `?c` is bound to the integer 40000. Before evaluating the `FILTER` clause, the result set contains `dbpedia:Milan_Cathedral`, but this result is discarded because the expression `40000 > 50000` evaluates to `FALSE`. Therefore, the query in [▶ Table 8.10](#) gives an empty result.

As a more complex example, consider the SPARQL query in [▶ Table 8.12](#) that returns resources whose label contains the string “duomo” ignoring if it is upper or lower case, by restricting values of strings using the `regex()` function. The flag “i” tells the engine to ignore the case.

Multiple filter clauses can be combined using logic connectors illustrated in [▶ Table 8.11](#). For instance, the query in [▶ Table 8.13](#) uses a complex filter that checks

■ **Table 8.10**

SPARQL query that matches resources whose capacity is more than 50000

```

1. PREFIX dbpprop: <http://dbpedia.org/property/>
2. SELECT ?s
3. FROM <http://dbpedia.org/data/Milan_Cathedral.rdf>
4. WHERE { ?s dbpprop:capacity ?c .
5.         FILTER (?c > 50000) }
```

■ **Table 8.11**

Grammar of the SPARQL operators for constructing constraints

Operator	Meaning	Operand type	Result type
<i>Logical Connectives</i>			
!A	Logical NOT	xsd:boolean	xsd:boolean
A B	Logical OR	xsd:boolean	xsd:boolean
A && B	Logical AND	xsd:boolean	xsd:boolean
<i>XPath Tests</i>			
A = B	Equal to	Any literal typed with XSD	xsd:boolean
A != B	Different From	Any literal typed with XSD	xsd:boolean
A < B	Less Than	Any literal typed with XSD	xsd:boolean
A <= B	Less or Equal Than	Any literal typed with XSD	xsd:boolean
A > B	Greater Than	Any literal typed with XSD	xsd:boolean
A >= B	Great or Equal Than	Any literal typed with XSD	xsd:boolean
<i>SPARQL Tests</i>			
A = B	Equal RDF Term	RDF Term	xsd:boolean
A != B	Different RDF Term	RDF Term	xsd:boolean
sameTerm(A, B)	Same RDF Term	RDF Term	xsd:boolean
langMatches(A, B)	Is the literal A written in language B?	A a simple literal B a language tag	xsd:boolean
regex(A, B, F)	Does the literal A match the regular expression B considering the flag F ^a ?	A a simple literal or a xsd:string B a regular expression F a flag (optional)	xsd:boolean
bound(A)	Is the variable A bound?	A a variable	xsd:boolean
isIRI(A)	Is A e IRI	RDF Term	xsd:boolean
isBlank(A)	Is A e blank node	RDF Term	xsd:boolean
isLiteral(A)	Is A e literal	RDF Term	xsd:boolean
<i>SPARQL Accessors</i>			
lang(A)	What language is A written in?	Simple Literal	Lang tag

Table 8.11 (Continued)

Operator	Meaning	Operand type	Result type
Datatype(A)	What datatype is A typed with?	Typed Literal	IRI
<i>XPath Arithmetic</i>			
A * B	Multiply	Numeric	Numeric
A / B	Divide	Numeric	Numeric ^b
A + B	Sum	Numeric	Numeric
A - B	Subtrack	Numeric	Numeric

^aThe regular expression language is defined in XQuery 1.0 and XPath 2.0 Functions and Operators Section 7.6.1 Regular Expression Syntax [81]

^bIf both the operands are xsd:integer, the result cannot be xsd:decimal

Table 8.12

SPARQL query that matches resources whose label includes the string Duomo

```
1. PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
2. SELECT ?s
3. FROM <http://dbpedia.org/data/Milan_Cathedral.rdf>
4. WHERE { ?s rdfs:label ?l .
5.         FILTER (regex(str(?l), "duomo", "i" ) }
```

Table 8.13

SPARQL query that matches resources whose label includes the string Duomo or whose capacity exceeds 50000

```
1. PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
2. PREFIX dbpprop: <http://dbpedia.org/property/>
3. SELECT ?s
4. FROM <http://dbpedia.org/data/Milan_Cathedral.rdf>
5. WHERE { ?s rdfs:label ?l .
6.         FILTER (regex(str(?l), 'Cathedral' ) || (?c > 50000 ) }
```

whether the label contains the string "cathedral" OR the capacity is more than 50,000. The boolean operator OR is represented as "||" (AND is represented as "&&"). As a result the variable ?s is bound to dbpedia:Milan_Cathedral.

Note that, both in [Tables 8.12](#) and [8.13](#), the variable ?l is cast to string by using the str() function, before invoking the regular expression function regex. This is because the matched RDF term (i.e., "Duomo di Milano"@it) is a string with the language tag "@it". The str() function is a constructor function that can be used to

■ **Table 8.14**

Casting operations that are always allowed (Y), never allowed (N), and dependent on the lexical value (M)

from \ To	string	float	double	decimal	integer	dateTime	boolean
string	Y	M	M	M	M	M	M
float	Y	Y	Y	M	M	N	Y
double	Y	Y	Y	M	M	N	Y
decimal	Y	Y	Y	Y	Y	N	Y
integer	Y	Y	Y	Y	Y	N	Y
dateTime	Y	N	N	N	N	Y	N
boolean	Y	Y	Y	Y	Y	N	Y
IRI	Y	N	N	N	N	N	N
literal	Y	M	M	M	M	M	M

cast from one datatype to another. ▶ [Table 8.14](#) summarizes the casting operations that are always allowed (Y), never allowed (N), and dependent on the lexical value (M). For example, a casting operation from “3.14”^{^^xsd:string} to xsd:float is possible, whereas from “π”^{^^xsd:string} to xsd:float is not.

8.1.7 More Sophisticated Graph Patterns

When clients issue SPARQL queries that use basic graph pattern, the entire pattern must match in order to have a solution. For instance, the query in ▶ [Table 8.15a](#) requests resources of type `dbcat:Churches_in_Milan` that have a label in English, a latitude, and a longitude. The query, once issued against the two RDF graphs in ▶ [Tables 8.2](#) and ▶ [8.3](#), matches `dbpedia:Basilica_of_Sant%27Ambrogio`. However, the same query does not match with `dbpedia:Milan_Cathedral`, because the RDF description of Milan Cathedral in ▶ [Table 8.2](#) does not include latitude and longitude.

However, due to the nature of the Web, it cannot be assumed that all RDF graphs have a regular and complete structure. SPARQL queries can return also partial solutions, if a pattern is expressed by using the OPTIONAL clause. When an *optional pattern* does not match, no RDF term is bound to the solution, but the solution is not eliminated. For instance, the query in ▶ [Table 8.15c](#), in which the latitude and a longitude pattern of the query in ▶ [Table 8.14a](#) are grouped within an optional pattern, returns the matching shown in ▶ [Table 8.15d](#). Note that a binding for `dbpedia:Milan_Cathedral` is present, even if the variables `?lat` and `?long` are not bound.

Table 8.15

OPTIONAL patterns (in query c) enables partial results to be found (in result d)

a)	<pre> 1. PREFIX skos: <http://www.w3.org/2004/02/skos/core#> 2. PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> 3. PREFIX dbcat: <http://dbpedia.org/resource/Category:> 4. PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos#> 5. SELECT ?s ?label ?lat ?long 6. FROM <http://dbpedia.org/data/Milan_Cathedral.rdf> 7. FROM <http://dbpedia.org/data/Basilica_of_Sant%27Ambrogio.rdf> 8. WHERE { ?s skos:subject dbcat:Churches_in_Milan . 9. ?s rdfs:label ?label . 10. FILTER langMatches(lang(?label), "EN") . 11. ?s geo:lat ?lat . 12. ?s geo:long ?long . }</pre>			
b)	?s	?label	?lat	?long
	dbpedia:Basilica_of_Sant%27Ambrogio	"Basilica of Sant 'Ambrogio"@en	45.4624	9.1758
c)	<pre> 1. PREFIX skos: <http://www.w3.org/2004/02/skos/core#> 2. PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> 3. PREFIX dbcat: <http://dbpedia.org/resource/Category:> 4. PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos#> 5. SELECT ?s ?label ?lat ?long 6. FROM <http://dbpedia.org/data/Milan_Cathedral.rdf> 7. FROM <http://dbpedia.org/data/Basilica_of_Sant%27Ambrogio.rdf> 8. WHERE { ?s skos:subject dbcat:Churches_in_Milan . 9. ?s rdfs:label ?label . 10. FILTER langMatches(lang(?label), "EN") . 11. OPTIONAL { 12. ?s geo:lat ?lat . 13. ?s geo:long ?long . } 14. }</pre>			
d)	?s	?label	?lat	?long
	dbpedia:Milan_Cathedral	"Milan Cathedral"@en	null	Null
	dbpedia:Basilica_of_Sant%27Ambrogio	'Basilica of Sant 'Ambrogio '@en	45.4624	9.1758

In addition to optional patterns, SPARQL provides the UNION keyword to express alternative graph patterns. Using UNION, the query processor tries matching several patterns. For example, the query in [Table 8.16](#) uses the UNION keyword to ask for Milan churches using either the category `Churches_in_Milan` in DBpedia or the concept `ChurchesInMilan` in the YAGO knowledge base [71].

Table 8.16

SPARQL query that uses the UNION keyword

```

1. PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
2. PREFIX dbcat: <http://dbpedia.org/resource/Category:>
3. PREFIX yago: <http://dbpedia.org/class/yago/>
4. SELECT ?s
5. FROM <http://dbpedia.org/data/Milan_Cathedral.rdf>
6. WHERE {
7.     { ?s skos:subject dbcat:Churches_in_Milan . }
8.     UNION
9.     { ?s skos:subject yago:ChurchesInMilan . }
10. }
```

8.1.8 Dealing with Blank Nodes

So far in this chapter, the notion of *blank node* was ignored; however, it plays an important role in the RDF data model. This session briefly recalls such a notion and explains how SPARQL deals with blank nodes.

Not all elements in RDF triples are IRI. The W3C working group allowed IRI to be substituted by anonymous resources, named *blank node*. Table 8.17a shows an example of blank node usage in modeling the events of Alice's life: she was born in 7.3.1975; she married in 24.6.1997; she divorced in 2003 (not reported in the data); and she married again in 21.2.2007. Instead of forcing one to design an IRI for each event in Alice's life, RDF allows using a blank node. Syntactically, a blank node is denoted by a special namespace represented by the character underscore "_"; for instance, _:e1, _:e2, and _:e4 are all blank nodes. Technically, this representation is named *label form*.

Being anonymous, blank nodes in some RDF serialization can be omitted. For instance, Table 8.17b shows an abbreviated form allowed by Turtle that uses square brackets (i.e., "[" and "]") instead of the label form. In Turtle syntax, they appear explicitly only if they are referred in some other part of the RDF graph serialization.

SPARQL allows querying RDF graphs containing blank nodes. Blank nodes in graph patterns act as non-distinguished variables and can be indicated by either the label form, such as "_:e", or the abbreviated form "[]". Table 8.18 shows a query that returns Alice's life events using a blank node in the label form (i.e., Table 8.18a) and in the abbreviated form (i.e., Table 8.18b).

Before closing this section, it is worth noting that the user could have issued a regular query with a variable "?e" instead of the label form "_:e" and the result would have been the same. Practitioners normally use blank node in graph pattern only in the abbreviated form to avoid using those variables that are not projected in the SELECT clause. Moreover, if the label form is used together with multiple graph patterns, the same blank node label cannot be used in two different graph patterns of the same query.

Table 8.17

An example of RDF graphs where blank nodes are used

a)	<pre>@prefix e: <http://purl.org/vocab/bio/0.1/> . :Alice e:event _:e1 . _e1 e:date "1975-07-03" . _e1 a e:Birth . Alice e:event _:e2 . _e2 e:date "1997-06-24" . _e2 a e:Marriage . Alice e:event _:e4 . _e4 e:date "2007-02-21" . _e4 a e:Marriage .</pre>
b)	<pre>@prefix e: <http://purl.org/vocab/bio/0.1/> . :Alice e:event [e:date '1975-07-03', a e:Birth .], [e:date '1997-06-24', a e:Marriage .] [e:date '2007-02-21', a e:Marriage .]</pre>

Table 8.18

Two equivalent SPARQL queries with blank nodes

a)	<pre>1. PREFIX : <http://ex.org/> 2. PREFIX e: <http://purl.org/vocab/bio/0.1/> 3. SELECT ?type ?date 4. WHERE { 5. :Alice e:event [6. a ?type ; 7. e:date ?date 8.] 9. }</pre>
b)	<pre>1. PREFIX : <http://ex.org/> 2. PREFIX e: <http://purl.org/vocab/bio/0.1/> 3. SELECT ?type ?date 4. WHERE { 5. :Alice e:event _:e . 6. _:e a ?type . 7. _:e e:date ?date . 8. }</pre>

8.1.9 Negation in SPARQL

The semantics of negation is not intuitive, and errors can be easily made. For instance, consider the data in [Table 8.19a](#): Alice likes Milano, Bob likes both Milano and New York, and Carl likes New York. A user may want to issue a SPARQL query that

Table 8.19

Errors that can be easily made by misunderstanding negation in SPARQL

a)	<pre>@prefix : <http://example>. :A :name "Alice" . :A :likes :Milano . :B :name "Bob" . :B :likes :Milano . :B :likes :NewYork. :C :name "Carl" . :C :likes :NewYork .</pre>
b)	<pre>1. PREFIX : <http://example> 2. SELECT ?name 3. WHERE { 4. ?person :name ?name . 5. ?person :likes ?x . 6. FILTER (?x != :Milano) 7. }</pre>
c)	<pre>?name "Bob" "Carl"</pre>

identifies the people who do not (*explicitly*) like Milano; in the example, the query returns Carl. In doing so, a user may be tempted to issue the query in [Table 8.19b](#).

[Table 8.19c](#) shows that the interpretation of such a query is not the intuitive one. Carl is in the result set, because he does not explicitly state that he likes Milano, and Alice is not in the result set, because she likes Milano. But why is Bob in the result set? For the SPARQL engine, the existence of the triple `<:B :likes :NewYork>` causes Bob's inclusion in the result set. Indeed the query in [Table 8.19](#) means “give me all the people who like any city different from Milano.”

In order to issue a query that produces the most natural meaning of negation, (i.e., one in which Bob is excluded from the result), a user should use *negation as failure*. The idea is to test that a graph pattern is *not* expressed by specifying an OPTIONAL graph pattern that introduces a variable and then test that such variable is not bound.

The query in [Table 8.20](#) correctly identifies the people that do not (*explicitly*) like Milano. The graph pattern at line 4 identifies all the people. The OPTIONAL graph pattern from line 5 to 8, matches any triple that describes if a person likes Milano using the variable `?x` as a place holder. Finally, the FILTER clauses at line 9 tests if the variable `?x` is not bound. Therefore, a person that matches the graph pattern at line 4 as well as the OPTIONAL graph pattern from line 5 to 8 will be discarded by the FILTER clauses at line 9. Only those people who do not explicitly like Milano remain in the variable binding.

Negation as failure is not been largely adopted among practitioners. As presented in [Sect. 8.4.1.4](#), SPARQL 1.1 working draft proposes to include the clause NOT EXIST in order to explicitly support negation.

Table 8.20

Query corresponding to the intuitive meaning of negation (uses negation as failure)

```

1. PREFIX: <http://example>
2. SELECT ?name
3. WHERE {
4.     ?person :name ?name .
5.     OPTIONAL {
6.         ?person :likes ?x .
7.         FILTER (?x = :Milano )
8.     }
9.     FILTER (! BOUND(?x) )
10. }
```

8.1.10 The FROM, FROM NAMED, and GRAPH Clauses

In the previous sections, the `FROM` clause is used to indicate the RDF graph against which the query is executed, without providing in-depth explanation of its usage. This section explains what a SPARQL engine does when it evaluates a `FROM` clause: it introduces the `FROM NAMED` clause and it describes how a SPARQL query can match different parts of the query pattern against different graphs using the `GRAPH` clause.

The first step in this construction is the introduction of *RDF dataset*. As explained in [Sect. 8.1.1](#), the RDF data model expresses information as graphs consisting of triples with subject, predicate, and object. An RDF *dataset* represents a collection of graphs: the *default graph*, which does not have a name, and zero or more *named graphs*, whose name is specified using an IRI. The RDF dataset is built by using `FROM` and `FROM NAMED` clauses:

- The `FROM` clause is used to identify the contents to be loaded in the default graph. Note that more than one `FROM` clause can appear; the IRIs, which follow the `FROM` clause, identify Internet locations from where data are read (and not the names of the graphs). The SPARQL endpoint dereferences each IRI, obtains a set of RDF graphs and merges them to form the default graph.
- The `FROM NAMED` clause is used to identify a named graph. The graph is given the name of the IRI that follows the `FROM NAMED` clause and the RDF statements are read from that location. Multiple `FROM NAMED` clauses cause multiple named graphs to be added to the dataset.

When querying a collection of graphs, the `GRAPH` clause allows matching patterns against named graphs. Also the `GRAPH` clause can be used in two ways.

- The `GRAPH` clause provides a mechanism to limit pattern matching to a given graph. The IRI http://dbpedia.org/data/Milan_Cathedral.rdf, shown in [Table 8.21a](#) at row 7, specifies the “Milan Cathedral” named graph, which is loaded with the `FROM NAMED` clause at row 4. The triple pattern at row 8 matches two triples in the “Milan Cathedral” graph and it does not match any triple in the “Basilica of

Table 8.21

Two SPARQL queries that show the usage of the FROM, FROM NAMED, and GRAPH clauses

a)	<pre> 1. PREFIX yago: <http://dbpedia.org/class/yago/> 2. PREFIX skos: <http://www.w3.org/2004/02/skos/core#> 3. SELECT ?monument ?category 4. FROM NAMED <http://dbpedia.org/data/Milan_Cathedral.rdf> 5. FROM NAMED <http://dbpedia.org/data/ Basilica_of_Sant%27Ambrogio.rdf> 6. WHERE { 7. GRAPH <http://dbpedia.org/data/Milan_Cathedral.rdf> 8. { ?monument skos:subject ?category } 9. 10. }</pre>
b)	<pre> 1. PREFIX yago: <http://dbpedia.org/class/yago/> 2. PREFIX skos: <http://www.w3.org/2004/02/skos/core#> 3. SELECT ?src ?monument 4. FROM NAMED <http://dbpedia.org/data/Milan_Cathedral.rdf> 5. FROM NAMED <http://dbpedia.org/data/ Basilica_of_Sant%27Ambrogio.rdf> 6. WHERE { 7. GRAPH ?src 8. { ?monument skos:subject yago:ChurchesInMilan } 9. }</pre>

Sant’Ambrogio” graph, even if matching triples are present, because the GRAPH clause limits the matching to the “Milan Cathedral” graph.

- In addition, the GRAPH clause can be followed by a variable which matches one of the IRIs of the named graphs in the dataset. For instance, the variable `?src` in [Table 8.21b](#) at row 7 matches the IRIs of the named graph in which the triple pattern at row 8 will also match. When the SPARQL engine executes the query against the data in [Table 8.2](#), it instantiates a mapping between the variable `?src` and the IRI http://dbpedia.org/data/Milan_Cathedral.rdf, because the Milan Cathedral RDF graph contains the triple `dbpedia:Milan_Cathedral skos:subject yago:ChurchesInMilan` (see [Table 8.2](#), row 9) that matches the triple pattern at row 8 of [Table 8.21](#).

Finally, note that the RDF graph identified in the FROM clause can be computed dynamically. Flickrwrapp [25] and Sindice [57] are good examples of Web applications that expose REST services returning results as RDF graphs.

Flickrwrapp is a REST service that accepts as input a geo-coordinate point p and a radius r , and it returns an RDF graph that includes links to Flickr images that were geo-tagged within a circle of radius r centered in the point p .

The query in [Table 8.22](#) shows how to extract the links to the Flickr photos from the RDF graph dynamically generated by Flickrwrapp when asking for Milan Cathedral geo-coordinates (i.e., latitude 45.464169 and longitude 9.191154) and a radius of 200 m.

Sindice is a lookup index over Semantic Web resources. It offers APIs that allows Semantic Web applications to automatically locate documents containing information about a given resource. As in the case of Flickrwrappr, Sindice APIs return RDF graphs. For instance, the advance search API allows sending a triple pattern such as the one shown in [▶ Table 8.23a](#), where the character “*” is a wildcard. The answer is an RDF graph of the kind shown in [▶ Table 8.23b](#).

■ **Table 8.22**

SPARQL query in which the FROM clause loads an RDF graph dynamically computed by a REST service

```
1. SELECT ?pic
2. FROM <http://www4.wiwiss.fu-berlin.de/
           flickrwrappr/data/
           photosDepictingLocation/45.464169/
           9.191154/250>
3. WHERE { ?x <http://xmlns.com/foaf/0.1/depiction> ?pic . }
```

■ **Table 8.23**

An example of the results in RDF format of Sindice Advance Search APIs

a)	* <http://www.w3.org/2000/01/rdf-schema#label>+"Milan cathedral"
b)	<pre><rdf:RDF> <Results rdf:about="..."> <terms> * <http://www.w3.org/2000/01/rdf-schema#label> "milan cathedral"b </terms> ... </Results> <ResultPage rdf:about="..."> <entry rdf:resource="#result1"/> <entry rdf:resource="#result2"/> ... </ResultPage> <Entry rdf:about="#result1"> <dc:title>Milan Cathedral, Mailänder Dom</dc:title> <link rdf:resource="http://dbpedia.org/resource/ Milan_Cathedral"/> ... <rank>1</rank> </Entry> ... </rdf:RDF></pre>

■ **Table 8.24**

A SPARQL query that uses **Sindice Advance Search APIs** in the **FROM** clause to retrieve the **DBpedia resource** that has **“Milan Cathedral”** as label

```

1. SELECT ?link
2. FROM <http://api.sindice.com/v2/search?q=+%3Chttp%3A%2F%
   2Fwww.w3.org%2F2000%2F01%2Frd-schema%23label%3E+%22milan
   +cathedral%22&qt=advanced&page=1>
3. WHERE {
4.     ?x <http://sindice.com/vocab/search#link> ?link .
5.     FILTER (regex(str(?link), "dbpedia" ) )

```

Using the Sindice Advance search API in a **FROM** clause, a SPARQL query (see [▶ Table 8.24](#)) can be written that retrieves the DBpedia resource that has “Milan Cathedral” as label.

SPARQL practitioners can, in this way, discover Semantic Web resources. This is an important step toward querying the entire Web with SPARQL (see [▶ Sect. 8.4.6](#) for more details on such a future opportunity).

8.1.11 Solution Modifiers

Modifiers alter the solution generated in evaluating the **WHERE** clause before sending the result to the query issuer. At a logical level, a modifier takes a solution as input, manipulates it, and generates a new solution as output. For instance, the **SELECT** clause (see [▶ Sect. 8.2.2](#)), which is a *projection modifier*, takes the solution generated by the **WHERE** clause and selects a subset of the variables for generating the final result.

At a physical level, modifiers rarely are implemented as described at logical level. For instance, only naïve implementation of the **SELECT** clause would be evaluated only after the **WHERE** clause. An optimized SPARQL implementation should perform algebraic optimizations: it should interleave the generation of the solution in the **WHERE** clause with the application of the projection modifier. It is beyond the scope of this chapter to discuss SPARQL optimizations; for a comprehensive discussion see [66].

Apart from the projection modifier, the solution modifiers available in SPARQL are:

- *Order modifier*, that is, the **ORDER BY** clause that puts the solutions in order by using a sequence of ordering clauses that are either ascending (indicated by the **ASC** () modifier) or descending (indicated by the **DESC** () modifier). **ASC** is assumed as default modifier. For an example, readers are referred to the query in [▶ Table 8.30](#) of [▶ Sect. 8.2.1.1](#).
- *Distinct and reduced modifier*, represented, respectively, by the **DISTINCT** and the **REDUCED** clause. A solution sequence with no **DISTINCT** clause will preserve duplicates. The **DISTINCT** clause eliminates all duplicates, whereas **REDUCED** is a ‘best effort’ **DISTINCT** that removes consecutive duplicates from the query result. For an example, readers are referred to the query in [▶ Table 8.31](#) of [▶ Sect. 8.2.1.2](#).

- *Limit* and *offset modifier*, represent, respectively, the `LIMIT` and the `OFFSET` clauses. The `LIMIT` and `OFFSET` clauses allow one to retrieve just a portion of the results that are generated by the query. The `LIMIT` clause indicates the maximum number of results that can be returned; the `OFFSET` clause indicates to skip a given number of results. For an example, readers are referred to the query in [▶ Table 8.34](#) of [▶ Sect. 8.2.2](#).

8.1.12 Query Forms `CONSTRUCT`, `ASK`, and `DESCRIBE`

The previous sections are focused on `SELECT` queries against RDF graphs, thereby obtaining as result, a set of variable bindings. However, as explained at the beginning of the section, innovative and interesting features of SPARQL are found as well in the other three query forms: `CONSTRUCT`, `ASK`, and `DESCRIBE`.

8.1.12.1 `CONSTRUCT` Form

The `CONSTRUCT` query form returns an RDF graph specified by the query designer as *graph template*. For instance, in [▶ Table 8.25a](#), the graph template at row 4 states that a church is an `ex:Big_Church` if it has a capacity larger than 30,000 people. The result, shown in [▶ Table 8.25b](#), is a triple constructed by substituting the variable `?s` in the graph template with the RDF term bound to it.

In the general case, the graph template contains multiple triples and the result of a `CONSTRUCT` query is an RDF graph that combines the triples, obtained as described above, into a single RDF graph by set union. If a variable is unbound, as in the case of an `OPTIONAL` group pattern, it does not contribute to forming any triple in the output RDF graph.

It is important to note that SPARQL `CONSTRUCT` queries can be used to generate new triples from existing ones. Seen from this viewpoint, SPARQL can be interpreted both as a data integration facility that allows one to translate from one vocabulary into another

▣ **Table 8.25**

SPARQL for constructing a triple stating that `Milan_Cathedral` has type `Big_Church`

a)	<pre> 1. PREFIX dbpprop: <http://dbpedia.org/property/> 2. PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> 3. PREFIX ex: <http://ex.org/resources/> 4. CONSTRUCT { ?s rdf:type ex:Big_Church . } 5. FROM <http://dbpedia.org/data/Milan_Cathedral.rdf> 6. WHERE { ?s dbpprop:capacity ?c . FILTER (?c > 30000) } </pre>
b)	<code>dbpedia:Milan_Cathedral rdf:type ex:Big_Church</code>

(see query in ▶ [Table 8.26](#)) and as a logical *rule language* that allows inferring new triples from existing ones (see query in ▶ [Table 8.27](#)).

The SPARQL query in ▶ [Table 8.26](#) illustrates how a CONSTRUCT query can be used to solve a data integration problem by adding links to an RDF repository following a given pattern: all resources that are categorized as `dbcat:Churches_in_Milan` should also be categorized as `dbcat:Churches_in_Milan`.

The query in ▶ [Table 8.27](#) illustrated how important ontological constructs, which are not available in the Web Ontology Language OWL [40], such as the role composition, can be easily declared by means of SPARQL CONSTRUCT query. As an example, consider the notion of *uncle* visually represented in ▶ [Table 8.27](#): Alice has Bob as parent and Bob has Carl as brother, hence Alice has Carl as uncle. The property `hasUncle` can be

■ **Table 8.26**

The SPARQL construct query (b) is used to transform the triple in (a) in the one in (c)

a)	1. <code>dbpedia:Milan_Cathedral skos:subject dbcat:Churches_in_Milan .</code> 2. <code>dbpedia:Basilica_of_Sant'Ambrogio skos:subject dbcat:Churches_in_Milan .</code>
b)	1. PREFIX <code>yago: <http://dbpedia.org/class/yago/></code> 2. PREFIX <code>skos: <http://www.w3.org/2004/02/skos/core#></code> 3. PREFIX <code>dbcat: <http://dbpedia.org/resource/Category:></code> 4. CONSTRUCT { <code>?s {?s skos:subject yago:ChurchesInMilan .}</code> } 5. WHERE { 6. <code>?s skos:subject dbcat:Churches_in_Milan .</code> 7. }
c)	1. <code>dbpedia:Milan_Cathedral skos:subject yago:ChurchesInMilan .</code> 2. <code>dbpedia:Basilica_of_Sant'Ambrogio skos:subject yago:ChurchesInMilan .</code>

■ **Table 8.27**

The SPARQL construct query (b) has the expressive power as the logical rule “uncle”

a)	1. <code>Alice hasParent Bob .</code> 2. <code>Bob hasBrother Carl .</code>	
b)	1. CONSTRUCT { <code>?x hasUncle ?z .</code> } 2. WHERE { <code>?x hasParent ?y .</code> 3. <code>?y hasBrother ?z .</code> }	
c)	1. <code>Alice hasUncle Carl .</code>	

modeled as a simple SPARQL CONSTRUCT query (see [▶ Table 8.27b](#)) that constructs the triple: `?x hasUncle ?y` if it matches the pattern: `?x hasParent ?y` and `?y hasBrother ?z`. When issued against the triples in [▶ Table 8.27a](#), such a query constructs the triple in [▶ Table 8.27c](#).

8.1.12.2 ASK Form

In several cases, a client is only interested in knowing if a given query can be answered by a SPARQL endpoint, without being interested in the actual result. The ASK query form covers such a need. When an ASK query is issued to a SPARQL processor, the processor answers whether or not a solution exists. For instance, the query in [▶ Table 8.28a](#), which is issued against the RDF data of Milano Cathedral in [▶ Table 8.2](#), returns `false`; whereas the query in [▶ Table 8.28b](#), which is issued against the RDF data of Sant’Ambrogio in [▶ Table 8.3](#), returns `true`.

8.1.12.3 DESCRIBE Form

Finally, a client may not know enough information about a resource to be able to issue a query. In those cases, it can still issue a SPARQL query using the DESCRIBE form. A DESCRIBE SPARQL query returns an RDF graph determined by the SPARQL endpoint. Such a query can have two forms: the very simple one shown in [▶ Table 8.29a](#) or the slightly more complex one shown in [▶ Table 8.29b](#). The first query is used to get information about a given resource (indicated via its URI), the second query gets information about those resources that satisfy a regular WHERE clause. Both of them, if issued against the entire DBpedia, return a description of Sant’Ambrogio church similar to the RDF data in [▶ Table 8.3](#).

In reality, by trying the query, the description will most likely not be exactly the same as the one in [▶ Table 8.3](#); the reason is twofold. First of all, Wikipedia evolves and DBpedia evolves as well; therefore, the information in [▶ Table 8.3](#) can become obsolete. Secondly, the

■ **Table 8.28**

SPARQL ASK query that checks whether a resource has latitude and longitude

a)	<pre>1. PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos#> 2. ASK 3. FROM <http://dbpedia.org/data/Milan_Cathedral.rdf> 4. WHERE { ?s geo:lat ?lat . 5. ?s geo:long ?long . }</pre>
b)	<pre>1. PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos#> 2. ASK 3. FROM <http://dbpedia.org/data/ Basilica_of_Sant%27Ambrogio.rdf> 4. WHERE { ?s geo:lat ?lat . 5. ?s geo:long ?long . }</pre>

■ **Table 8.29**

Two SPARQL DESCRIBE queries returning a description of Sant’Ambrogio church

a)	1. DESCRIBE <http://dbpedia.org/resource/Basilica_of_Sant%27Ambrogio>
b)	1. PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos#> 2. PREFIX xsd: <http://www.w3.org/2001/XMLSchema#> 3. DESCRIBE ?s 4. WHERE { ?s geo:lat "45.4624"^^xsd:float . 5. ?s geo:long "9.1758"^^xsd:float . }

Schema	Example
http://<domain>/<endpoint>?<parameters> where <parameters> can include: <ul style="list-style-type: none"> • query=<encoded query string> • default-graph-uri=<encoded graph URI> • named-graph-uri=<encoded graph URI> 	http://dbpedia.org/sparql? query= SELECT+%3Fs+%0D%0AWHERE+{%3F ... &default-graph-uri= http%3A%2F%2Fdbpedia.org%2Fdata ... &named-graph-uri= http%3A%2F%2Fdbpedia.org%2Fdata ...
NOTE: <i>default-graph-uri</i> and <i>named-graph-uri</i> can occur zero or more times	

■ **Fig. 8.3**

An illustration of how a SPARQL query can be encoded in an HTTP GET using the SPARQL protocol

returned RDF graph is fully determined by the SPARQL endpoint; therefore, two different endpoints, which use DBpedia as default graph, can return different descriptions.

8.1.13 SPARQL Protocol

SPARQL is not only a query language, but it is also a simple communication protocol that a client can use to issue SPARQL queries against some *endpoint* receiving back answers. The SPARQL protocol is described abstractly with WSDL 2.0 [12] as a Web Service interface, with an associated HTML and a SOAP binding for such interface. Those descriptions are mainly dedicated to SPARQL engine developers; interested readers can refer to [13]. ▶ *Figure 8.3* shows the usage of the SPARQL protocol when associated to HTTP and using the GET method; the schema on the left side of the figure is exemplified on the right side.

The SPARQL protocol will most likely drastically change in future versions of SPARQL, because it does not respect several Web principles. A RESTful [24] approach is most likely to be followed in future versions of the SPARQL protocol. For more details, interested readers can refer to ▶ *Sect. 8.4.3*, where details about the ongoing developments are provided.

8.1.14 SPARQL Semantics

The features of SPARQL, taken one by one, are simple to describe and to understand. However, the combination of such features makes SPARQL a complex language, whose semantics were not completely understood when the language was initially specified in 2005. It has been shown [42] that the operational semantics of SPARQL given in the 2006 Working Draft [44] did not cover all the complexities brought by the specified constructs, and it included ambiguities, gaps, and features difficult to understand. The rest of this section describes the semantics of SPARQL query language using the algebraic representation proposed by [42] and currently described in the SPARQL W3C Recommendation [45] and in [43].

A central concept of such an algebraic representation is the basic graph pattern (BGP). In order to calculate the solution mapping of a BGP against an RDF graph, a SPARQL engine has to map all the variables in the BGP (the set V) into the terms in the RDF graph (the set T). A *term* can be an IRI, a blank node, or a literal. A solution μ is a partial function $\mu : V \rightarrow T$. A solution is valid if μ maps V in a subgraph of the RDF graph against which the query is evaluated. For instance, the solution μ that maps the variable x in the term t , the variable y in the term s , and the variable z in the term r , is:

$$\mu(?x \rightarrow t, ?y \rightarrow s, ?z \rightarrow r) = \{(?x, t), (?y, s), (?z, r)\}.$$

The solution of a BGP is a bag of mapping, an unordered set of solutions that admits duplicates. For instance, given the solutions μ_1 , μ_2 , and μ_3 :

$$\mu_1 = \{(?x, t_1), (?y, s_1), (?z, r_1)\}$$

$$\mu_2 = \{(?x, t_1), (?y, s_2), (?z, r_2)\}$$

$$\mu_3 = \{(?x, t_1), (?y, s_1), (?z, r_1)\}$$

the bag Ω that contains them is

$$\Omega(\mu_1, \mu_2, \mu_3) = \{\{(?x, t_1), (?y, s_1), (?z, r_1)\}, \{(?x, t_1), (?y, s_2), (?z, r_2)\}, \{(?x, t_1), (?y, s_1), (?z, r_1)\}\}$$

Note that the bag Ω contains both μ_1 and μ_3 even if they represent the same solution. Also note that the solutions are not ordered.

To provide the formal semantics of SPARQL constructs beyond BGP, four algebraic operators of BGP should be introduced:

- **Filter(*expr*, *pattern*):** this operator filters the *pattern* in input. It gives formal semantics to the following block in a WHERE clause:

```
WHERE { pattern
        FILTER (expr) }
```

The operator evaluates the expression $expr$ for the solution of $pattern$ and it gives as a result the solution of the $pattern$ if $expr$ evaluates true, otherwise it gives no results. Formally, if the solution of the $pattern$ is Ω , the filter operation returns:

$$Filter(expr, \Omega) = \{\mu | \mu \in \Omega \wedge expr(\mu) = true\}.$$

- **Join($pattern_1$, $pattern_2$):** this operator computes the join between the two patterns in input. It gives formal semantics to the following block in a WHERE clause:

```
WHERE { {pattern1}
        {pattern2}
```

To give the semantics of the join operator, some terminology must be introduced. Denoting with $dom(\mu)$ the domain of the solution μ (i.e., the set of the variable V in μ), two solutions μ_1 and μ_2 are *compatible* if:

$$\forall v \in (dom(\mu_1) \cap dom(\mu_2)) : \mu_1(v) = \mu_2(v)$$

The *insiemistic union* of the respective mappings variable-term of two compatible solutions is denoted as:

$$merge(\mu_1, \mu_2) = \mu_1 \text{ set } - \text{ union } \mu_2.$$

Given these two definitions, if the solutions of $pattern_1$ and $pattern_2$ are Ω_1 and Ω_2 , the join operator returns:

$$Join(\Omega_1, \Omega_2) = \{merge(\mu_1, \mu_2) | \mu_1 \in \Omega_1, \mu_2 \in \Omega_2 \wedge \mu_1, \mu_2 \text{ compatible}\}.$$

- **Union($pattern_1$, $pattern_2$):** this operator computes the union of the two patterns in input. It gives formal semantics to the following block in a WHERE clause:

```
WHERE { {pattern1}
        UNION {pattern2}
```

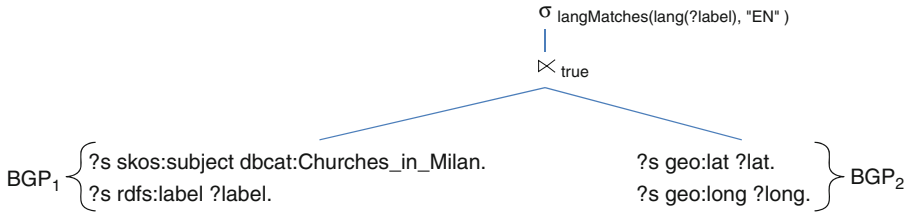
If the solutions of $pattern_1$ and $pattern_2$ are Ω_1 and Ω_2 , the union operator returns:

$$Union(\Omega_1, \Omega_2) = \{\mu | \mu \in \Omega_1 \vee \mu \in \Omega_2\}.$$

Duplicates in Ω_1 or Ω_2 are not removed; if μ appears n_1 times as solution of Ω_1 and it appears n_2 times as solution of Ω_2 , then μ appears n_1+n_2 times in the union.

- **LeftJoin ($pattern_1$, $pattern_2$, $expr$):** this operator computes the left-join (or semi-join) between the two patterns in input evaluating the expression $expr$. It gives formal semantics to the following block in a WHERE clause:

```
WHERE { {pattern1}
        OPTIONAL
        {pattern2}
        FILTER(expr) }
```



■ Fig. 8.4

A tree-based representation of the SPARQL query in [Table 8.15c](#)

To give the formal semantics of this operator, the auxiliary operator **Diff**(**pattern**₁, **pattern**₂, **expr**) has to be defined. Such an operator computes the insiemistic difference between the two patterns while evaluating the expression *expr*. If the solutions of *pattern*₁ and *pattern*₂ are Ω_1 and Ω_2 , this operator returns:

$$\text{Diff}(\Omega_1, \Omega_2, \text{expr}) = \{\mu \mid \mu \in \Omega_1 : \forall \mu' \in \Omega_2 : \mu, \mu' \text{ are not compatible} \\ \vee (\mu, \mu' \text{ compatible} \wedge \text{expr}(\text{merge}(\mu, \mu')) = \text{false})\}.$$

Given the definition of the *Diff* operator, the *LeftJoin* operator returns:

$$\text{LeftJoin}(\Omega_1, \Omega_2, \text{expr}) = \text{Filter}(\text{expr}, \text{Join}(\Omega_1, \Omega_2)) \cup \text{Diff}(\Omega_1, \Omega_2, \text{expr})$$

Note that the bag of solutions returned by the *Filter* and *Diff* operators are disjoint, thus they cannot generate duplicates.

Using these four operators, any graph pattern specified in a *WHERE* clause can be represented in an algebraic form. To complete the description of the semantics of SPARQL, a formal definition of the *SELECT* clause and the *DISTINCT*, the *ORDER BY*, the *LIMIT*, and the *OFFSET* modifiers should be provided. However, a suitable combination of the previous operators builds a table of values of variables that can be modified by applying the projection operation to express the *SELECT* clause, and extended relational algebra operations to express distinct, order, limit, and offset.

Putting in practice what has been described above, the algebraic definition of the SPARQL query in [Table 8.15c](#) is *Filter*(*LangMatches*(*lang*(*?label*), "EN"), *Leftjoint*(*BGP*₁, *BGP*₂, *true*)), where *BGP*₁ denotes the two initial patterns (see row 18 and 19) and *BGP*₂ denotes the two optional patterns (see row 22 and 23).

The SPARQL algebraic representation allows presenting a query as a tree. [Figure 8.4](#) shows the tree representation of the query in [Table 8.15c](#). For compactness, a symbol inspired by the relational algebra can be introduced for each operator. σ_{pred} denotes the *Filter* operator, \cup denotes the *Union* operator, \bowtie denotes the *Join* operator, \ltimes_{pred} denotes the *LeftJoin* operator, and $-_{\text{pred}}$ denotes the *Diff* operator.

8.2 Example Applications

This section provides examples of applications of SPARQL. It starts, in [Sect. 8.2.1](#), with “early days” applications, when RDF data were lacking and the Semantic Web

practitioners applied semantic technologies to bibliographic and conference data. ▶ [Section 8.2.2](#) moves on to “first uptakes” in the area of bioscience, the most relevant scientific community that adopted Semantic Web technologies early. Then, ▶ [Sects. 8.2.3](#), ▶ [8.2.4](#), and ▶ [8.2.5](#) are dedicated to interesting SPARQL queries that nowadays can be addressed to interlinked RDF repositories of music-related information, of common knowledge, and of governmental data.

8.2.1 Early Days

As with many other new technologies, aiming at the replacement of existing ones, in the early days of Semantic Web it was difficult to find any RDF repository. Actually, from 2001 to 2004, very few data sources were available and Semantic Web research was conducted on private datasets. The Semantic Web practitioners were criticized because they were not “eating their own dog food.”

The efforts of the Semantic Web community, toward reversing this allegation, brought three notable results:

- A SPARQL endpoint to query the DBLP Bibliography Database [[19](#)]
- A set of 25 SPARQL endpoints that expose the databases of major publishers, aggregators, and funding agencies in Computer Science, forming the back-end of RKB Explorer [[52](#)]
- A SPARQL endpoint that gives access to the Semantic Web Conference Corpus, also known as “the Semantic Web Dog Food Corpus” [[55](#)]

8.2.1.1 DBLP Bibliography Database

The DBLP database provides bibliographic information on major journals and conferences in the database and logic programming community. Currently, it includes about 800,000 articles of about 400,000 authors. In 2006, the Web-based Systems Group of the Freie Universität Berlin built a SPARQL endpoint (<http://www4.wiwiw.fu-berlin.de/dblp/sparql>) for querying the DBLP database through a D2R Server [[16](#)], in order to demonstrate that such a large relational database could be published as RDF data for usage related to the Semantic Web. In 2008, the L3S Research Center at the Leibniz Universität Hannover extended the initial DBLP D2R server; they will keep data up-to-date, because such a new version is also the basis for their Faceted DBLP [[23](#)] search engine.

▶ [Table 8.30](#) shows a query on the DBLP SPARQL endpoint, asking for the titles of the papers of Tim Berners-Lee, which are of type article or proceedings, ordered by date; such a query is formulated in ▶ [Table 8.30](#). Lines 1–4 indicate the prefixes for the vocabularies: `foaf` for the Friend of a Friend vocabulary [[26](#)], which includes the property name to indicate the name of a person; `dc` for the Dublin Core vocabulary [[21](#)], which includes the

■ **Table 8.30**

Example of query that can be issued against the DBLP SPARQL endpoint (<http://www4.wiwiss.fu-berlin.de/dblp/sparql>)

```

1. PREFIX foaf: <http://xmlns.com/foaf/0.1/>
2. PREFIX dc: <http://purl.org/dc/elements/1.1/>
3. PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
4. PREFIX dblp: <http://www4.wiwiss.fu-berlin.de/dblp/terms.rdf#>
5. SELECT ?date ?title
6. WHERE { ?author foaf:name "Tim Berners-Lee".
7.         ?paper dc:creator ?author.
8.         ?paper dc:date ?date.
9.         ?paper dc:title ?title.
10.        { ?paper rdf:type dblp:Article. }
11.        UNION {?paper rdf:type dblp:InProceedings. }
12. }ORDER BY DESC (?date)

```

properties `date` and `title` to indicate the publishing date and the title of a document; and `dblp` for the schema of DBLP database, which includes the datatypes `Article` and `InProceedings`, to indicate the respective types of articles. Line 5 indicates that the publication date and the title of the articles are variables to be matched. Lines 6–9 contain the pattern-matching part of the query, where line 6 is the one that selects the resource that identifies Tim Berners-Lee and lines 7–9 extracts the data of the paper. The `UNION` clause is used to connect the two alternative graph patterns that match either papers of type “article” or papers of type “in proceedings.” Finally, the `ORDER BY` clause is used to order the result set by date, and the `DESC ()` modifier puts most recent papers first.

8.2.1.2 RKB Explorer

The ReSIST “Network of Excellence” European Project started from the DBLP work described in the previous section and extended it by harvesting data from a number of major metadata resources [52]. Extensions covered the major publishers and aggregators in Computer Science, including Citeseer, ACM, and selected IEEE conferences, thus harvesting about 37 million triples. Other extensions covered the databases of Europe and US funding agencies, for additional 14 million triples. So far, 25 sources can be queried using SPARQL.

For instance, the <http://cordis.rkbexplorer.com/sparql> endpoint can be used for b. Such a query can be formulated in SPARQL as shown in [Table 8.31](#). Lines 1 and 2 indicate the prefixes for the two vocabularies from the Advance Knowledge Technologies (AKT) Project. Line 3 asks the name and the phone number of project leaders – and given that a single person may coordinate multiple projects, the clause `DISTINCT` is used to eliminate duplicates. Lines 4–5 match areas of interest whose name contains “information

■ **Table 8.31**

An example of query that it is possible to issue against one of the RKB Explorer repository of CORDIS projects (<http://cordis.rkbexplorer.com/sparql>)

```

1. PREFIX akts: <http://www.aktors.org/ontology/support#>
2. PREFIX akt: <http://www.aktors.org/ontology/portal#>
3. SELECT DISTINCT ?name ?tel
4. WHERE { ?areaOfInterest akts:has-pretty-name ?areaLabel .
5.         FILTER regex(str(?areaLabel), "information systems") .
6.         ?project akt:addresses-generic-area-of-interest ?
           areaOfInterest .
7.         ?project akt:has-project-leader ?projectLeader .
8.         ?projectLeader akt:full-name ?name .
9.         ?projectLeader akt:has-telephone-number ?tel .
10.        FILTER regex(str(?tel), "+39-02") .
11. }
12. LIMIT 10

```

systems”); in particular, at line 5 a `FILTER` clause with a regular expression (i.e., the `regex()` function) is used to perform the selection. Then, line 6 matches the projects in the area of interest, line 7 matches the project leaders of such projects, lines 8 and 9 match, respectively, their name and phone number, and line 10 uses a regular expression in a `FILTER` clause to select only those matches where the phone number contains the string “+39-02.” Finally at line 12, a `LIMIT` clause is used to retrieve only the first 10 results. This last clause allows getting some result in a few seconds; computing the entire query result takes several minutes.

8.2.1.3 Semantic Web Dog Food Corpus

Since 2006, the International, European, and Asian Semantic Web Conference series have maintained a SPARQL endpoint as well as a user interface [55] that give access to papers that were presented, people who attended, schedules, and other resources related to those conferences and colocated workshops.

For instance, the Semantic Web Dog Food endpoint (<http://data.semanticweb.org/sparql>) can be used for querying the names of the researchers that got a paper accepted at ISWC 2006. Given that Semantic Web Dog Food traces the provenance of RDF data and all ISWC 2006 data are grouped in a single graph, such a query can be formulated in SPARQL using the `GRAPH` clause as shown in ▶ [Table 8.32](#) at line 5. Such a clause forces the SPARQL engine to limit the scope of the query to the named graph. As in the query presented in ▶ [Table 8.32](#), given that a researcher can be author of multiple papers, a `DISTINCT` clause is used in line 3 to eliminate duplicates.

Table 8.32

An example of query that asks the names of the researchers that got a paper accepted at ISWC 2006

```
1. PREFIX foaf: <http://xmlns.com/foaf/0.1/>
2. PREFIX swrc: <http://swrc.ontoware.org/ontology#>
3. SELECT DISTINCT ?person ?name
4. WHERE {
5.     GRAPH <http://data.semanticweb.org/conference/iswc/2006/complete> {
6.         ?person foaf:name ?name.
7.         ?paper swrc:author ?person
8.     }
9. }
```

8.2.2 First Uptakers

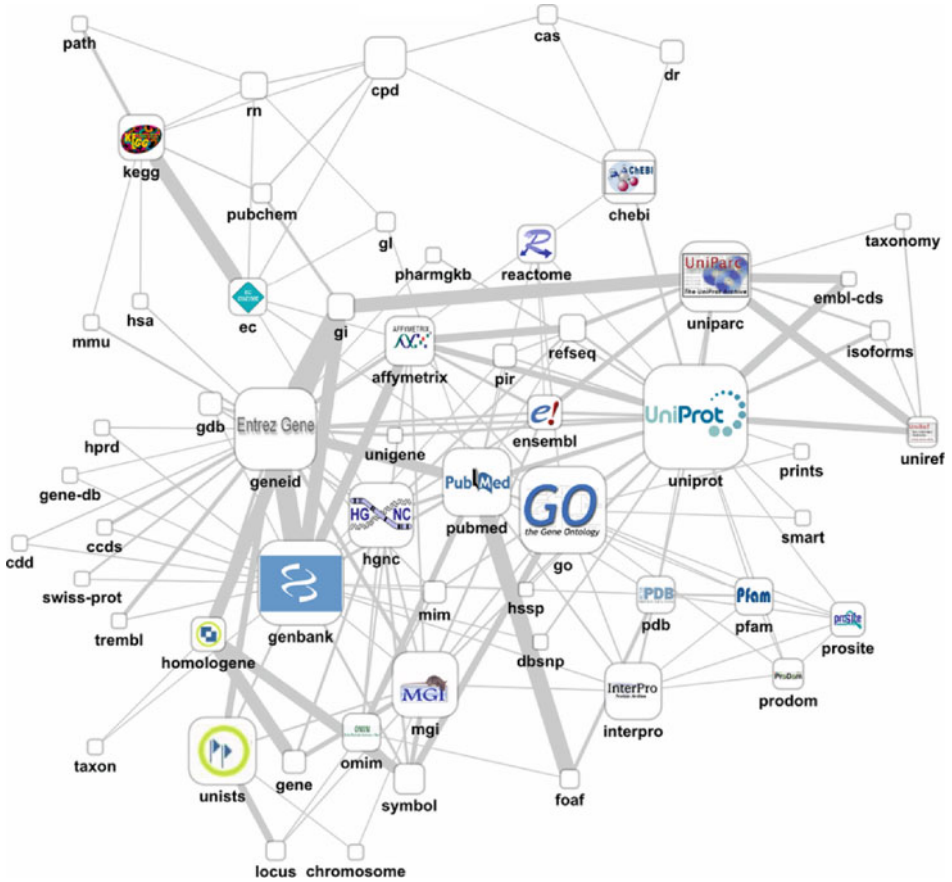
Every year, several organizations publish new bioinformatics databases on the Web. The most recent compilation of such a Web server, reported from the Web Server Issue of *Nucleic Acid Research* published in July 2009 [38], counted over 1,000 servers. With such a proliferation of knowledge sources, the bioinformatics community realized a long time ago the need for a global multi-site search engine and good data integration tools; a machinery that goes beyond the traditional data warehouse approach [9]. Such an approach, as a matter of fact, offers a solution to the problem, but it has at least three drawbacks:

1. Before being able to issue a query against such data warehouse, all the data have to be collected into a central repository and be integrated.
2. Keeping replicas up-to-date is difficult, especially in a context where the warehouse is maintained by an organization with no authority on those organizations who maintain the data sources.
3. If a database is not yet integrated in the data warehouse, accessing such a database is impossible.

The authors of the Bio2RDF project [6] observed that “A system that would be able to query different databases available on the Internet would solve that problem. Indeed, this is one goal of the Semantic Web: to offer the data warehouse experience without moving the data into a central repository.”

The Bio2RDF project created a common ontology and normalized URIs that allow issuing SPARQL queries against 42 SPARQL endpoints [8]. Each SPARQL endpoint wraps using Virtuoso or D2R a public available bioinformatics database for a total of 140 gigabytes of XML and text data, which are equivalent to 46 million RDF documents.

► *Figure 8.5* offers a visual representation of the Bio2RDF interlinked set of SPARQL endpoints; nodes represent SPARQL endpoints (where the node size visually represents



■ Fig. 8.5

The interlinked set of SPARQL endpoints made available by the bio2rdf project (image source: wiki of the bio2rdf project)

the number of RDF data in the repository) and the links represent cross-references among the SPARQL endpoints (where the link thickness visually represents the number of cross-references). ▶ [Table 8.33](#) gives a short description of some of those endpoints, including the ones used in the rest of this section. A complete list of all the endpoints made available by the Bio2RDF project is available on Freebase [8].

Current SPARQL specifications do not give to clients the possibility to issue a complex query over a network of SPARQL endpoints. Therefore, the Bio2RDF project proposes the following strategy [7]:

1. Write the SPARQL query as if a global repository were available.
2. Split the query in a set of CONSTRUCT queries that can be issued against the existing SPARQL endpoint in order to fetch the data.

■ **Table 8.33**

Some of the 42 bioinformatics databases exposed as SPARQL endpoints by the Bio2RDF project

Name	Description
KEGG Pathway database	<p>KEGG PATHWAY is a collection of manually drawn pathway maps representing one's knowledge on the molecular interaction and reaction networks for: metabolism, genetic information processing, environmental information processing, cellular processes, and human diseases. It also includes structure relationships (KEGG drug structure maps) in drug development.</p> <p>Website: http://www.genome.jp/kegg/pathway.html SPARQL endpoint: http://kegg.bio2rdf.org/sparql</p>
GeneID	<p>Entrez Gene is NCBI's database for gene-specific information. It focuses on the genomes that have been completely sequenced, that have an active research community to contribute gene-specific information, or that are scheduled for intense sequence analysis.</p> <p>Website: www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=gene SPARQL endpoint: http://geneid.bio2rdf.org/sparql</p>
Open Biomedical Ontologies	<p>A virtual library of biomedical ontologies, including ICD, the UMLS and its incorporated terminologies, and the content of the Open Biomedical Ontologies Library.</p> <p>Website: http://www.bioontology.org/ SPARQL endpoint: http://obo.bio2rdf.org/sparql</p>
PubMed	<p>PubMed is a service of the US National Library of Medicine that includes over 18 million citations from MEDLINE and other life science journals for biomedical articles, dating back to 1948. PubMed includes links to full text articles and other related resources.</p> <p>Website: http://www.ncbi.nlm.nih.gov/pubmed/ SPARQL endpoint: http://pubmed.bio2rdf.org/sparql</p>
UniProt	<p>Universal Protein Resource (UniProt) is the union of the Swiss-Prot, TrEMBL, and PIR protein database activities. It provides a central resource for querying protein sequences and functional annotations by making use of the three database components mentioned above, each addressing a key need in protein bioinformatics.</p> <p>Website: http://www.uniprot.org/ SPARQL endpoint: http://uniprot.bio2rdf.org/sparql</p>
DailyMed	<p>DailyMed provides high quality information about marketed drugs. It provides health information providers and the public with a standard, comprehensive, up-to-date, lookup of medication content and labeling, as found in medication package inserts.</p> <p>Website: http://dailymed.nlm.nih.gov/ SPARQL endpoint: http://www4.wiwiss.fu-berlin.de/dailymed/sparql</p>
DrugBank	<p>The DrugBank database is a bioinformatics and cheminformatics resource that combines detailed drug data (i.e., chemical, pharmacological and pharmaceutical) with comprehensive drug target information (i.e., sequence, structure, and pathway).</p> <p>Website: http://www.drugbank.ca/ SPARQL Endpoint: http://www4.wiwiss.fu-berlin.de/drugbank/sparql</p>
FlyAtlas	<p>FlyAtlas is a database of the Drosophila gene expression. It gives a quick answer to the question: "where is gene X expressed/enriched in the adult fly?"</p> <p>Website: http://flyatlas.org/ SPARQL Endpoint: http://openflydata.org/query/flyatlas_20080916</p>

3. Instantiate a local RDF store.
4. Issue the original SPARQL query against the local triple store.

The Bio2RDF service offers several REST services that adopt such a strategy. The basic one is the search service, which uses the URI schema <http://bio2rdf.org/search/<searchedTerm>>. For instance, a biologist who wants to search all SPARQL endpoints for enzyme “hexokinase” can dereference the URI <http://bio2rdf.org/search/hexokinase>. The implementation of such a search service is straightforward; the REST service issues the SPARQL CONSTRUCT query in [▶ Table 8.34](#) against all the SPARQL endpoints known to the Bio2RDF project, merges the results, and sends them to the client. Note that such a SPARQL query uses the `bif:contains` custom function of the SPARQL implementation of Virtuoso, which extends the standard `regex` function.

At the W3C HCLS face-to-face meeting in April 2009, the Bio2RDF team demonstrated the possibilities offered by the platform [7] asking: “which Gene Ontology terms describes KEGG pathway for mouse?” (see [▶ Table 8.35](#)). Answering such a query requires inspecting three SPARQL endpoints: KEGG pathways, GeneID, and Open Biomedical Ontologies.

[▶ Tables 8.36](#), [▶ 8.37](#), and [▶ 8.38](#) show three queries that, suitably orchestrated, retrieve the triples that are necessary in order to answer the global query in [▶ Table 8.35](#).

The CONSTRUCT SPARQL query in [▶ Table 8.36](#) operates as follows:

- Line 6 and 7 match mouse pathways by accepting only pathway URIs that include the string “mmu” (standing for “mouse”).

■ Table 8.34

Internal query that is sent by the Bio2RDF search service to all SPARQL endpoints when asked to dereference the URI <http://bio2rdf.org/search/hexokinase>

```

1. PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2. PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3. PREFIX dc: <http://purl.org/dc/elements/1.1/>
4. PREFIX bio: <http://bio2rdf.org/ns/bio2rdf:>
5. PREFIX bios: <http://bio2rdf.org/ns/search:>
6. CONSTRUCT {
7.   <http://bio2rdf.org/search/hexokinase>
8.     rdf:type bio: SearchResults;
9.     bios:hasResult ?s; rdfs:SeeAlso ?s .
10.   ?s rdfs:label ?label; dc:title ?title .
11.   ?s ?p ?o . }
12. WHERE {
13.   ?s ?p ?o .
14.   FILTER(bif:contains(?o, "hexokinase"))
15.   OPTIONAL { ?s rdfs:label ?label . }
16.   OPTIONAL { ?s dc:title ?title . }
17. } LIMIT 2000

```

■ **Table 8.35**

SPARQL query: “which Gene Ontology term describes KEGG pathway for mouse?” (source: Bio2RDF demonstration at W3C HCLS face-to-face meeting in April 2009)

```

1. SELECT ?keggPathwayName ?geneName ?geneOntologyTerm
2. WHERE {
3.   ?keggPathway a <http://bio2rdf.org/ns/kegg#Pathway> .
4.   ?keggPathway rdfs:label ?keggPathwayName .
5.   ?keggPathway ?p ?gene .
6.   ?gene <http://bio2rdf.org/ns/bio2rdf#xGO> ?go .
7.   ?gene rdfs:label ?geneName .
8.   ?go <http://bio2rdf.org/ns/go#name> ?geneOntologyTerm .
9. }

```

■ **Table 8.36**

Get the list of genes from KEGG pathways of mouse

```

1. CONSTRUCT {
2.   ?keggPathway a <http://bio2rdf.org/ns/kegg#Pathway> .
3.   ?keggPathway <http://bio2rdf.org/ns/bio2rdf#Pathway2Gene> ?gene .
4. }
5. WHERE {
6.   ?keggPathway a <http://bio2rdf.org/ns/kegg#Pathway> .
7.   FILTER (regex(?keggPathway, "mmu") .) .
8.   ?keggPathway <http://bio2rdf.org/ns/kegg#xRelation> ?xRelation .
9.   ?xRelation ?p1 ?entry .
10.  FILTER (?p1 = <http://bio2rdf.org/ns/kegg#xEntry1>
11.    || ?p1 = <http://bio2rdf.org/ns/kegg#xEntry2> ) .
12.  ?entry <http://bio2rdf.org/ns/kegg#xRef> ?keggGene .
13.  ?keggGene <http://www.w3.org/2002/07/owl#sameAs> ?gene .
14. }

```

■ **Table 8.37**

The query that can be used to get the list of genes from GeneID and the list of pathways from KEGG

```

1. CONSTRUCT {
2.   ?s ?p ?o .
3. }
4. WHERE {
5.   ?s ?p ?o .
6.   ?s a ?type .
7.   FILTER (   ?type = <http://bio2rdf.org/ns/kegg#Pathway>
8.             || ?type = <http://bio2rdf.org/ns/geneid#Gene> )
9. }

```

■ **Table 8.38**

Get the GO annotations for each gene

```

1. CONSTRUCT {
2.   ?go <http://bio2rdf.org/ns/go#name> ?name .
3. }
4. WHERE {
5.   ?go <http://bio2rdf.org/ns/go#name> ?name .
6. }

```

- Lines from 8 to 12 get the list of genes related to such pathways.
- Line 13 also matches the triples from the GeneID database that, by using the `owl:sameAs` property, are in correspondence with genes in the KEGG database. (The query in [▶ Table 8.36](#) assumes that the SPARQL endpoint operates under simple RDF entailment (see [▶ Sect. 8.4.5](#)). The pattern at line 13 only matches triples stating that an ID of a gene in Kegg dataset is `owl:sameAs` an ID of a gene in another dataset. Note that a SPARQL endpoint working under OWL-DL entailment regime is required in order to match triples that assert equality in the inverse direction.)
- Lines from 1 to 4 construct the relationships between pathways and genes.

The triples constructed by this query match the patterns at lines 3 and 5 of the global query in [▶ Table 8.35](#).

In order to construct the triples that will match the patterns at lines 4, 6, and 7 of the query in [▶ Table 8.35](#), the CONSTRUCT SPARQL query in [▶ Table 8.37](#) is next issued against both the GeneID and the KEGG SPARQL endpoints. The possibility to issue the same query against the two heterogeneous endpoints is achieved through the usage of the FILTER clause at lines 7 and 8.

The pattern at line 8 of [▶ Table 8.35](#), which matches the human readable names of the Gene Ontology terms, is the only part of the global query that is not expressed yet. For such a part, the SPARQL query in [▶ Table 8.38](#) can be issued against the Open Biomedical Ontologies SPARQL endpoint.

By merging the results of the queries in [▶ Tables 8.36](#), [▶ 8.37](#), and [▶ 8.38](#) into one local RDF repository and by issuing the global query in [▶ Table 8.35](#) against such a repository, it is finally possible to give an answer to the question: “which Gene Ontology terms describe KEGG pathway for mouse?”

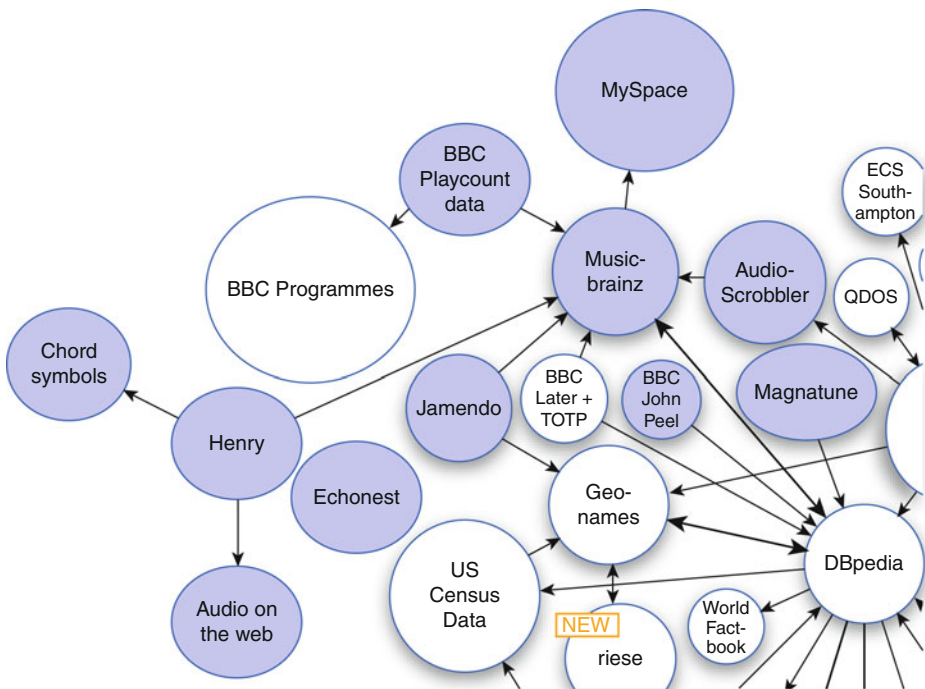
A set of other independent initiatives complement the Bio2RDF project in exposing existing bioinformatics databases as SPARQL endpoint. [▶ Table 8.33](#) describes three of them.

8.2.3 Larger Audience Applications: DBTune Case Study

The trend toward the integration of disparate databases, exposed as SPARQL endpoints, extends as well to other fields. Music is an interesting case study, because the music

industry of the 1980s, led by blockbusters, was completely changed first by the Internet and then by the Web 2.0. The first big change was caused by song sharing systems such as Napster, when millions of people started to be able to download songs on a planetary scale and to publish their own music. Then, a movement toward musical niches, populated by garage bands and obscure musicians, found fertile soil in the Web 2.0 online communities, such as MySpace, and in many cases reached economic sustainability thanks to Web 2.0 music suggesting services, such as Pandora, Last.FM, and Apple, selling individual music tracks at 1\$ on iTunes. The music industry today is populated by a large number of small production labels and individuals that publish music directly on the Internet.

The Music Ontology community [37] and the DBTune project [20] together offer services for the music industry that are similar to Bio2RDF for bioinformatics. The Music Ontology provides main concepts and properties for describing music (i.e., artists, albums, tracks, but also performances, arrangements, etc.); DBTune hosts a number of SPARQL endpoints and linked data servers, providing access to 14 billion RDF triples describing music-related structured data. ▶ *Figure 8.6* reports a diagram showing as grey circles the datasets available on DBTune and as edges their connection with some other datasets currently available on the Semantic Web. Three datasets of DBTune are exposed as SPARQL endpoints: MusicBrainz, Jamendo, and BBC playcount data.



■ **Fig. 8.6**

The diagram shows as grey circles the datasets available on DBTune (source: DBTune)

8.2.3.1 MusicBrainz

The MusicBrainz Project dataset is the central node of DBTune. MusicBrainz (<http://musicbrainz.org/>) aims at being an open-source system for looking up audio CDs on the Internet. DBTune exposed the data as a SPARQL endpoint, which allows one to access more than 36 million triples, including links to 51,000 DBpedia pages and 15,000 MySpace artist profiles.

The query in [▶ Table 8.39](#) demonstrates the possibility to navigate links between different datasets. The pattern at line 4 matches the identifier of “Alanis Morissette” in MusicBrainz starting from the one in DBpedia. Then, the pattern at line 5 matches all the resources connected to such MusicBrainz identifier using the property `owl:sameAs`. Potentially, this pattern matches also URIs that do not belong to the MySpace dataset. Given that MySpace identifiers are URI of the kind <http://dbtune.org/myspace/<artist>>, a FILTER clause can impose that the result only includes MySpace identifiers; as shown at line 6. the URI can be converted in a string using the `str()` function and the presence of the string “myspace” can be verified via the regular expression function `regex()`.

8.2.3.2 Jamendo

Jamendo (<http://jamendo.org/>) is a large repository of Creative Commons licensed music, based in France. DBTunes exposes it as a SPARQL endpoint including links to the GeoNames dataset, a worldwide geographical database available free under a Creative Commons license. For instance, a client can issue the Jamendo SPARQL endpoint (<http://dbtune.org/jamendo/sparql/>) the following query: “select artists within Jamendo who made at least one album that was tagged as “folk” by a Jamendo user, sorted by the number of inhabitants of the places where such artists are located.” The SPARQL query formulation is shown in [▶ Table 8.40](#).

■ **Table 8.39**

Query showing the linking of DBpedia, MusicBrainz, and MySpace datasets, as supported by SPARQL endpoint on DBTune (<http://dbtune.org/musicbrainz/sparql>)

```

1. PREFIX owl: <http://www.w3.org/2002/07/owl#>
2. PREFIX dbp: <http://dbpedia.org/resouce/>
3. SELECT ?MySpaceArtistID WHERE {
4.   ?MusicBrainzArtistID owl:sameAs dbp:Alanis_Morissette .
5.   ?MusicBrainzArtistID owl:sameAs ?MySpaceArtistID .
6.   FILTER regex(str(?MySpaceArtistID), "myspace") .
7. }
```

8.2.3.3 BBC Playcount Data

BBC made available (<http://mashed-audioandmusic.dyndns.org/>) a dataset of play counts of artists per episode and brands in their program catalog (<http://www.bbc.co.uk/programmes>), for the Mashed 2008 event (<http://mashed08.backnetwork.com/>). DBTune provides a SPARQL endpoint (<http://dbtune.org/bbc/playcount/sparql/>) in order to access these play counts, as well as links to MusicBrainz and BBC programs. Three main

■ **Table 8.40**

Query for the Jamendo SPARQL endpoint on DBTune (<http://dbtune.org/jamendo/sparql/>)

```

1. PREFIX geo: <http://www.geonames.org/ontology#>
2. PREFIX wgs: <http://www.w3.org/2003/01/geo/wgs84_pos#>
3. PREFIX mo: <http://purl.org/ontology/mo/>
4. PREFIX foaf: <http://xmlns.com/foaf/0.1/>
5. PREFIX tags: <http://www.holygoat.co.uk/owl/redwood/0.1/tags/>
6. SELECT DISTINCT ?artistName ?placeName ?population
7. WHERE {
8.   ?artist a mo:MusicArtist .
9.   ?artist foaf:based_near ?place .
10.  ?artist foaf:name ?artistName .
11.  ?artist foaf:made ?album .
12.  ?album tags:taggedWithTag <http://dbtune.org/jamendo/tag/folk> .
13.  ?place geo:name ?placeName .
14.  ?place geo:population ?population .
15. } ORDER BY ?population

```

■ **Table 8.41**

Query for the BBC play count SPARQL endpoint on DBTune

```

1. PREFIX mo: <http://purl.org/ontology/mo/>
2. PREFIX po: <http://purl.org/ontology/po/>
3. PREFIX pc: <http://purl.org/ontology/playcount/>
4. PREFIX foaf: <http://xmlns.com/foaf/0.1/>
5. PREFIX dc: <http://purl.org/dc/elements/1.1/>
6. SELECT ?brand ?title ?count
7. WHERE {
8.   ?artist a mo:MusicArtist .
9.   ?artist foaf:name "The Beatles" .
10.  ?pc pc:object ?artist .
11.  ?pc pc:count ?count .
12.  FILTER (?count>10) .
13.  ?brand a po:Brand .
14.  ?brand pc:playcount ?pc .
15.  ?brand dc:title ?title
16. }

```

ontologies are needed to represent these data: the Music Ontology [37], the BBC Programmes Ontology [4], and a small ontology to represent play counts. The BBC Programmes Ontology complements the Music Ontology with a simple vocabulary for describing programs, which covers brands, series (seasons), episodes, broadcast events, and broadcast services. The play count ontology defines the play count concept.

For instance, the query in ▶ [Table 8.41](#) issued against the BBC play count SPARQL endpoint, returns the BBC bands in which “The Beatles” were featured at least ten times.

8.2.4 Addressing “The Modigliani Test”

This chapter opened with a reference to a Richard MacManus’s post on ReadWriteWeb called the “Modigliani Test,” that is, the search on Web-scattered RDF data of the locations of all the original paintings of Modigliani. The information available in the Semantic Web allowed Atanas Kiryakov, in April 2010 [31], to formulate the SPARQL query shown in ▶ [Table 8.42](#). Such a query *partially* passes the “Modigliani Test,” as it answers all the locations of Modigliani’s painting currently available in DBPedia and FreeBase, which are only few.

Such SPARQL query (see ▶ [Table 8.42](#)) can be issued to the Linked Data Semantic Repository (LDSR) endpoint (<http://www.freebase.com/>). LDSR was created by Kiryakov’s company Ontotext by collecting, cleaning up, and indexing DBPedia, Freebase, Geonames, UMBEL (<http://www.umbel.org/>), and WordNet (<http://wordnet.princeton.edu/>) into a single semantic repository in a way, which allows those to be queried and used in a reliable fashion. Simply enough, lines from 10 to 13 express the graph patterns that get the owners of *all* Modigliani’s paintings known in Freebase. However, three different graph patterns (respectively, at line 14, 15, and 18) are necessary to retrieve the locations from DBPedia and Freebase. Requiring these three OPTIONAL patterns shows the expertise necessary to formulate a SPARQL query and, thus, the long way to run before the *futuristic search engine* imagined by MacManus could be made available to final Web users. ▶ [Table 8.42b](#) shows the results of the query as in August 2010.

8.2.5 Future Applications

The US and UK governments may become important data providers for future applications. Groups such as the Guardian [28] and the Sunlight Foundation (<http://www.sunlightfoundation.com/>) have been pushing for a long time for free access to data collected with public funds. Government data have already been published by using various technologies, including Semantic Web ones in websites like watchdog.net, mysociety.org, and govtrack.us. Two announcements by the US and UK governments

Table 8.42

Query for the “Modigliani test” on the LDSR endpoint

a)	<pre> 1. PREFIX fb: <http://rdf.freebase.com/ns/> 2. PREFIX dbpedia: <http://dbpedia.org/resource/> 3. PREFIX dbp-prop: <http://dbpedia.org/property/> 4. PREFIX dbp-ont: <http://dbpedia.org/ontology/> 5. PREFIX umbel-sc: <http://umbel.org/umbel/sc/> 6. PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> 7. PREFIX ot: <http://www.ontotext.com/> 8. SELECT DISTINCT ?painting_l ?owner_l ?city_fb_con ?city_db_loc ?city_db_cit 9. WHERE { 10. ?p fb:visual_art.artwork.artist dbpedia:Amedeo_Modigliani ; 11. fb:visual_art.artwork.owners [12. fb:visual_art.artwork_owner_relationship.owner ?ow 13.] ; 14. ot:preferredLabel ?painting_l . 15. ?ow ot:preferredLabel ?owner_l . 16. OPTIONAL{ ?ow fb:location.location.containedby [ot:preferredLabel ?city_fb_con] } . 17. OPTIONAL{ ?ow dbp-prop:location ?loc . 18. ?loc rdf:type umbel-sc:City ; 19. ot:preferredLabel ?city_db_loc } 20. OPTIONAL{ ?ow dbp-ont:city [ot:preferredLabel ?city_db_cit] } 21. }</pre>				
b)	painting_l	owner_l	city_fb_con	city_db_loc	city_db_cit
	Head	Museum of Modern Art	New York City		
	Anna Zborowska	Museum of Modern Art	New York City		
	Reclining Nude	Museum of Modern Art	New York City		
	Portrait of Diego Rivera	The São Paulo Museum of Art		São Paulo	
	Woman with a Necklace	School of the Art Institute			Chicago
	Portrait of a Woman	School of the Art Institute			Chicago
	Madam Pompadour	School of the Art Institute			Chicago
	Jeanne Hébuterne	Barnes Foundation	Philadelphia		

seem to turn such a move toward the adoption of Semantic Web technologies, including SPARQL, into a concrete possibility:

- President Barack Obama, on January 21st, 2009, stated [77]: “Transparency promotes accountability and provides information for citizens about what their Government is doing [. . .] My Administration will take appropriate action, consistent with law and policy, to disclose information rapidly in forms that the public can readily find and use. Executive departments and agencies should harness new technologies to put information about their operations and decisions online and readily available to the public.”
- The UK Prime Minister, on June 10th, 2009, announced [76]: “So that Government information is accessible and useful for the widest possible group of people, I have asked Sir Tim Berners-Lee, who led the creation of the World Wide Web, to help us drive the opening up of access to Government data in the Web over the coming months.”

As Tim Berners-Lee writes in [74]: “this, 2009, is the year for putting government data online.”

8.3 Related Resources

Several SPARQL implementations exist. [▶ Table 8.43](#) provides pointers to websites, licenses, download pages, and documentation pages for some of the SPARQL implementation available. A longer list is maintained at <http://esw.w3.org/topic/SparqlImplementations>.

D2R and Virtuoso are tools for publishing relational databases on the Semantic Web. Both of them allow clients to query a relational database using SPARQL and enable RDF and HTML browsers to navigate the content of the database. Requests from the Web are rewritten into SQL queries via mappings performed at configuration time. This on-the-fly translation allows for the publishing of RDF from large databases and eliminates the need for replicating the data into a dedicated RDF repository.

4store, AllegroGraph, ARQ, Sesame, ARC, Open Anzo, Taslis, and RDF::Query are all native SPARQL engines that allow their clients to query an RDF repository. In most of the cases, an intermediate virtualization layer sits between the RDF repository and the storage solution. In this manner, the same RDF repository can be implemented over both a custom storage engine and standard relational databases. Custom storage engines are best to have a better query performance, but inserting and updating triples in such engines takes longer than in engines built over relational databases.

[▶ Figure 8.7](#) provides a vision of the standard compliance of the seven SPARQL implementations [65]; the numbers in the cells represent percentages of tests in the DAWG test suite [18] that were correctly performed by each SPARQL implementation. This table shows that ARQ, Sesame, and RDF::Query are fully compliant with

■ **Table 8.43**

Some implementations of SPARQL

Name	Description
4store	Website: http://4store.org/ License: GPL Language: PHP, Ruby, Python, and Java Clients are available Download page: http://4store.org/trac/wiki/Download Documentation page: http://4store.org/trac/wiki/Documentation
AllegroGraph	Website: http://www.franz.com/agraph/allegrograph/ License: open-source and commercial Language: Java, c#, Python, Lisp Download page: http://www.franz.com/agraph/downloads/ Documentation page: http://www.franz.com/agraph/support/documentation/current/agraph-introduction.html
ARC	Website: http://arc.semsol.org/ License: W3C software license, GPL v2 and GPL v3 Language: php Download page: http://arc.semsol.org/download Documentation page: http://arc.semsol.org/docs
ARQ	Website: http://jena.sourceforge.net/ARQ/ License: open-source with some limitation Language: Java Download page: http://jena.sourceforge.net/ARQ/download.html Documentation page: http://jena.sourceforge.net/ARQ/documentation.html
D2R Server	Website: http://www4.wiwiw.fu-berlin.de/bizer/d2r-server/ License: GPL Language: Java Download page: http://sourceforge.net/projects/d2rq-map/files/D2R%20Server/
Open Anzo	Website: http://www.openanzo.org License: Eclipse Public License -v 1.0 Language: Java, Javascript and .NET Download page: http://www.openanzo.org/downloads.html Documentation page: http://www.openanzo.org/projects/openanzo/wiki
RDF::Query	Website: http://www.openanzo.org License: GPL Language: Perl Download page: http://search.cpan.org/dist/RDF-Query/
Sesame	Website: http://www.openrdf.org License: BSD-style Language: Java Download page: http://www.openrdf.org/download.jsp Documentation page: http://www.openrdf.org/documentation.jsp
Talis Platform	Website: http://www.talis.com/platform/ License: does not apply, because the platform is available as SaaS Language: any, the platform is available as a set of REST APIs Download page: none, the platform is available as SaaS Documentation page: http://n2.talis.com/wiki/Main_Page
Virtuoso	Website: http://virtuoso.openlinksw.com/ License: GPL v2 and Commercial Language: Java and .NET Download page: http://oplweb.openlinksw.com:8080/download/virtuoso.vsp Documentation page: http://docs.openlinksw.com/virtuoso/

SPARQL specifications, while the other implementations fail to be compliant to some degree.

8.4 Future Issues

The current specifications of SPARQL (Query Language, Protocol, and Result Format) were published by the RDF Data Access Working Group [79] in January 2008. Even before the specification reached the Recommendation status, SPARQL was widely implemented and deployed. The community of SPARQL practitioners expresses several needs for extensions, which most likely will be handled in version 1.1 of SPARQL [32]; some of them are already covered, albeit not in standard form, and have now a large implementation basis, to the extent that users can already provide feedbacks relative to the various alternative implementations.

The rest of this section provides an overview of the proposed extensions and of future trends in SPARQL. [▶ Section 8.4.1](#) presents several foreseen extensions to the query language that include aggregates, subqueries, projections, and negation. [▶ Section 8.4.2](#) is dedicated to updating RDF graphs through SPARQL. [▶ Section 8.4.3](#) describes two current efforts for going beyond the SPARQL 1.0 protocol. [▶ Section 8.4.4](#) describes vocabularies for describing SPARQL endpoints. [▶ Section 8.4.5](#) describes how SPARQL can operate under different entailment regimes. [▶ Section 8.4.6](#) presents two proposals for querying the entire Semantic Web with SPARQL. Finally, [▶ Sect. 8.4.7](#) sketches proposals for extending SPARQL to the management of RDF streams. All sections contain references to the literature about the topic under discussion and, whenever possible, to the most recent release of SPARQL 1.1 Working Drafts.

8.4.1 Extensions to the Query Language

8.4.1.1 Aggregates

Current specifications of SPARQL lack operations for grouping results and calculating aggregates. For instance, SPARQL does not allow counting the number of results and calculating min, max, and average of sets or bags of numeric values. These functions are an important feature of SQL, which also allows one to group rows with the `GROUP BY` clause and to filter the resulting groups with the `HAVING` clause. Given the practical importance of aggregates, several SPARQL implementations support them.

OpenLink Virtuoso proposes a SPARQL extension that adds `COUNT`, `COUNTDISTINCT`, `MAX`, `MIN`, and `AVG` functions to the `SELECT` clause [3]. With the count aggregate, the argument may be either the `**` character, meaning counting all rows, or a variable name, meaning counting all the rows where this variable is bound. In addition to this set of standard SQL aggregate functions, Virtuoso provides a method to create

Feature	Allegro Graph	Open Anzo	ARC	ARQ	Sesame	RDF::Query	Virtuoso
ASK query form	74%	44%	53%	100%	100%	100%	76%
Basic graph pattern matching. Triple pattern constructs.	87%	75%	67%	100%	100%	100%	73%
CONSTRUCT query form	100%	100%	100%	100%	100%	100%	100%
Core bits of SPARQL. Prefixed names, variables, etc.	91%	77%	68%	100%	100%	100%	71%
FILTER clauses and expressions	83%	58%	55%	100%	100%	100%	72%
OPTIONAL pattern matching	83%	100%	80%	100%	100%	100%	88%
RDF datasets. Default and named graphs. GRAPH keyword	84%	100%	14%	100%	100%	100%	23%
SELECT query form	92%	83%	71%	100%	100%	100%	69%
Sorting (ORDER BY) and slicing (LIMIT, OFFSET)	100%	100%	90%	100%	100%	100%	77%
UNION pattern matching	82%	100%	46%	100%	100%	100%	38%
Compliance with SPARQL Grammar	partial	full	Partial	Full	full	full	partial

■ Fig. 8.7

Compliance to SPARQL specification of several SPARQL implementations (Source: W3C [SPARQLCompliance])

■ **Table 8.44**

Comparison of the Virtuoso and ARC2 SPARQL extensions for aggregates

Virtuoso
1. SELECT ?predicate count (?object) 2. WHERE {?subject ?predicate ?object}
ARC2
1. SELECT ?predicate COUNT(?object) AS ?count 2. WHERE {?subject ?predicate ?object. } 3. GROUP BY ?predicate

application-specific aggregate functions [15]. All variables appearing in aggregate functions implicitly group results; therefore no explicit GROUP BY clause is added to SPARQL. There is no explicit syntax for the SQL HAVING clause in Virtuoso SPARQL, but the clause can be implemented by using subqueries (see 🔗 Sect. 8.4.1.2).

ARQ proposes instead to add both GROUP BY and HAVING clauses to SPARQL syntax; it supports COUNT, COUNT DISTINCT, and SUM aggregate functions [2]. This is also the choice of ARC2, which extends SPARQL SELECT clause with COUNT, MAX, MIN, AVG, and SUM and requires the GROUP BY clause [1]. Simpler extensions, mostly limited to COUNT, are also present in Glitter, Garlik's JXT, and Dave Beckett's Redland.

➤ *Table 8.44* shows how to count the number of objects for each distinct predicate in a graph using the Virtuoso and the ARC2 SPARQL extensions for aggregates.

The query language extensions under discussion for SPARQL 1.1 [62] include aggregates. The current Working Draft proposes built-in calls for SUM, AVG, COUNT, COUNT DISTINCT, MIN, and MAX; an AS expression to assign a name to the result of a call to a built-in; a GROUP BY clause to partition the result set; and a HAVING clause to state additional conditions. Most likely, the FILTER keyword would be used in place of HAVING.

8.4.1.2 Subqueries

Nesting the results of a query within another query is sometimes needed, for instance for selecting elements only when their counting satisfies filter predicates; in this case, a subquery could be used for counting and filtering such elements, which could then be further used within a higher-order query. Currently, SPARQL practitioners need first to issue a query, then to parse the results with a query-specific script, and then to launch a second query; with a provision for subqueries, a single query would be sufficient. Three different nesting possibilities can be foreseen:

1. SELECT subqueries substituting for a group pattern
2. ASK subqueries used in a FILTER clause as boolean condition
3. CONSTRUCT or DESCRIBE subqueries used in a FROM or FROMNAMED clause

Currently both Virtuoso [70] and ARQ [69] support the first kind of nesting. [▶ Table 8.45](#) shows an example of nested select query that finds goods with more than ten orders.

The query language extensions under discussion for SPARQL 1.1 [62] include subqueries of the kind supported by Virtuoso and ARQ. Such subqueries are the same as top-level SELECT queries, but they cannot have the prologue (i.e., PREFIX) and the dataset description clauses (i.e., FROM and FROM NAMED).

8.4.1.3 Project Expressions

A commonly requested feature is the ability to project arbitrary expressions out of the query. For instance, the query in [▶ Table 8.46](#) matches the given name and the surname of a person and then projects the full name obtained by concatenating name and surname using the XPath-Functions `fn:string-join()`.

Further extensions that can be considered as project expressions include:

- Computing arithmetic expressions upon numeric variables, such as the product of `?unit_weight` and `?quantity` returning the total weight
- Parsing strings, such as the use of `fn:substring(?url, 8, fn:string-length(?url))` to eliminate the `http://` part of the URL

■ Table 8.45

Example of nested SELECT query; the subquery substitutes for a group pattern

```

1. PREFIX ex: <http://example/>
2. SELECT ?x WHERE {
3.   ?x a ex:Good .
4.   { SELECT ?x ( count(?order) as ?q )
5.     WHERE { ?x ex: order ?order }
6.     GROUP BY ?x }
7.   FILTER ( ?q > 10 )
8. }
```

■ Table 8.46

Example of usage of a project expression

```

1. PREFIX foaf: <http://xmlns.com/foaf/0.1/>
2. SELECT fn:string-join(?givenName, ' ', ?surname) AS ?fullName
3. WHERE {
4.   ?person foaf:givenname ?givenName .
5.   ?person foaf:surname ?surname .
6. }
```

- Encoding strings as an URI, such as `fn:encode-for-uri("100% organic")`
- Getting the current date and time, by using the function `fn:current-dateTime()`

Several SPARQL implementations support project expressions with slightly different syntax. In ARQ [69] and Glitter, the `AS` keyword inside the brackets indicates an expression. In Virtuoso, the `AS` keyword is optional, and parentheses around the expression are required.

The query language extensions under discussion for SPARQL 1.1 [62] include project expressions using, optionally, the `AS` keyword to give a name to the result of the expression evaluation.

8.4.1.4 Negation

In SPARQL, it is not easy to write queries that identify all resources that do “not” have a certain property. Examples of such queries are “identify all people that do *not* have a particular skill,” “all papers that do *not* have a reviewer,” “all customers that did “not” buy a given good,” etc. Checking the absence of triples is a form of negation, called “negation as failure.”

As described in [Sect. 8.2.3](#), negation as failure can be expressed using a combination of `OPTIONAL`, `FILTER`, and `!BOUND`; see, for instance, the query in [Table 8.47](#), which identifies all the artists in Jamendo that have an album tagged as “folk” music, but not tagged as “pop” music.

The SPARQL practitioners found this syntax complex and difficult to learn. The pre-SPARQL languages RDF::Query [48] and SeRQL [54] offer two alternative syntaxes that express negation as failure using, respectively, the `UNSAID` and the `MINUS` operator. The `NOT EXIST` alias of the `UNSAID` operator is also present in ARQ SPARQL extensions [39]. [Table 8.48](#) shows how to formulate the query of [Table 8.47](#) using `NOT EXISTS`.

Table 8.47

Example of query that combines `OPTIONAL`, `FILTER`, and `!BOUND` to express negation

```

1. PREFIX foaf: <http://xmlns.com/foaf/0.1/>
2. PREFIX tags: <http://www.holygoat.co.uk/owl/redwood/0.1/tags/>
3. SELECT ?artist
4. WHERE {
5.   ?artist foaf:made ?album .
6.   ?album tags:taggedWithTag <http://dbtune.org/jamendo/tag/folk> .
7.   OPTIONAL {
8.     ?album tags:taggedWithTag ?tag .
9.     FILTER (?tag = <http://dbtune.org/jamendo/tag/pop> ) .
10.  }
11.  FILTER (!BOUND(?tag)) .
12. }
```

Table 8.48**Example of query that uses NOT EXISTS extension to SPARQL to express negation**

```
1. PREFIX foaf: <http://xmlns.com/foaf/0.1/>
2. PREFIX tags: <http://www.holygoat.co.uk/owl/redwood/0.1/tags/>
3. SELECT ?artist
4. WHERE {
5.   ?artist foaf:made ?album .
6.   ?album tags:taggedWithTag <http://dbtune.org/jamendo/tag/folk> .
7.   NOT EXISTS { ?album tags:taggedWithTag
8.     <http://dbtune.org/jamendo/tag/pop> . }
```

The query language extensions under discussion for SPARQL 1.1 [62] include the NOT EXIST operator. It can be used in graph patterns as well as in FILTER expressions. An alternative design for negation based on the MINUS operator has also been discussed.

8.4.1.5 Other Extensions Under Discussion

Three other major extensions are under investigation: path expressions or property chains, in order to specify hierarchical property structures; custom filter functions, in order to extend the language filtering power; and query federation expressions, in order to partition a query over several SPARQL endpoints. All of them are research-oriented extensions that need further investigation, implementation, and user evaluation.

8.4.2 Insert, Update, and Delete

SPARQL does not provide any facility to add, update, or remove triples from a graph; for such a purpose, clients currently use implementation-specific APIs of the RDF store handling RDF graphs. In contrast, SQL allows one to perform INSERT, UPDATE, and DELETE operations, and therefore the community of SPARQL practitioners perceives the need for SPARQL extensions for manipulating RDF graphs in the same way.

The most comprehensive proposal is the SPARQL/Update language [67]. This language is a W3C member submission of July 15th, 2008, it is currently implemented in ARQ and Virtuoso, and it is under standardization as SPARQL 1.1 Update language [64]. Both SPARUL and SPARQL 1.1 updates support the following features:

- Insert new triples to an RDF graph.
- Delete triples from an RDF graph.
- Perform a group of update operations as a single action.
- Create a new RDF Graph and add it to a Graph Store.
- Delete an RDF graph from a Graph Store.

■ **Table 8.49**

Example of SPARQL extension for inserting triples in an RDF graph

```
1. PREFIX dc: <http://purl.org/dc/elements/1.1/>
2. INSERT DATA {
3.   <http://example/book3> dc:title "A new book" .
4.   <http://example/book3> dc:creator "A.N.Other" .
5. }
```

■ **Table 8.50**

Example of SPARQL extension for deleting triples from an RDF graph

```
1. PREFIX dc: <http://purl.org/dc/elements/1.1/>
2. PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
3. DELETE { ?book ?p ?v }
4. WHERE {
5.   ?book dc:date ?date .
6.   FILTER ( ?date < "2000-01-01T00:00:00"^^xsd:dateTime ) .
7.   ?book ?p ?v .
8. }
```

The two following examples, taken from the SPARQL/Update W3C member submission, illustrate some of the features of SPARQL/Update that will be present in SPARQL 1.1 update. The query in [Table 8.49](#) adds two RDF triples to the default graph of the RDF store.

The Query in [Table 8.50](#) deletes from the default graph all the records of books that were published before year 2000.

The SPARUL proposal does not include a protocol, because numerous criticisms emerged about the usage of SOAP as protocol for SPARQL (see also [Sect. 8.1.10](#)). At the time SPARUL was submitted, a RESTful [24] implementation of a protocol for SPARQL and SPARUL were under investigation and no clear picture had emerged yet. The SPARQL 1.1 working group is currently discussing these issues and an extended SPARQL protocol, which encompasses also updates, is under specification (see [Sect. 8.4.3](#) for more details).

8.4.3 Beyond the SPARQL 1.0 Protocol

As anticipated in [Sect. 8.1.10](#), the SPARQL protocol [13] will most likely drastically change in future versions of SPARQL, mainly because it is not RESTful [24]. The SPARQL Working Group at W3C is currently editing two independent working drafts: SPARQL 1.1 Protocol [61] and SPARQL 1.1 Uniform HTTP Protocol for Managing RDF Graphs [60]. Both documents are primarily intended for software developers interested in implementing SPARQL query and update services and clients. Hereafter, only a brief introduction to the two specifications is given; interested readers can directly check the full details in the two working drafts.

8.4.3.1 SPARQL Protocol

SPARQL 1.1 protocol mainly extends SPARQL 1.0 protocol to include SPARQL update. It defines an abstract interface as well as two HTTP bindings for a protocol to issue SPARQL Query and SPARQL Update statements against a SPARQL endpoint. Such a protocol is described at a high level of abstraction using WSDL 2.0 as a Web service. The WSDL 2.0 description includes an interface, two operations, and two faults. The interface describes the protocol at an abstract level. Querying and updating are the two possible operations. Both operations accept as input a message composed by a string – the query or the update statements – and zero or more RDF dataset descriptions. The two faults capture the error states caused by a malformed query submitted to a SPARQL endpoint and a SPARQL endpoint refusing to process the submitted query.

Note that while the specification uses WSDL 2.0, there is no obligation for implementers to use any WSDL library or Web service framework, as the HTTP bindings could instead be used. There are two HTTP bindings: one using GET and one using POST. For SELECT and CONSTRUCT queries, the GET binding is the preferred one, while the POST binding should be used only if the query, when encoded as an URL, exceeds practical limits. For SPARQL update queries, the POST binding must be used in accordance with HTTP usage principles. In each of these bindings, the two faults described in the abstract interface (i.e., malformed query and query request refused) are bound to HTTP status codes 400 *Bad Request* and 500 *Internal Server Error*, respectively [30].

8.4.3.2 Uniform HTTP Protocol for Managing RDF Graphs

SPARQL 1.1 protocol does not address the main criticism about SPARQL 1.0: it is not RESTful [24]. For this reason, the working group is also specifying a complementary RESTful protocol for managing RDF Graphs.

Such a protocol uses HTTP for remotely managing RDF datasets. It provides two kinds of operations:

1. Operations for creating, replacing, and removing RDF graphs
2. Operations for retrieving and adding RDF statements to existing RDF graphs

The HTTP *PUT* method is used to replace the content of an existing RDF graph identified by the requested IRI with the RDF statements in the payload of the HTTP *PUT* method. If the requested IRI does not point to an existing RDF graph, an RDF graph identified by the requested IRI is created and the RDF statements in the payload are uploaded in the newly created graph. The HTTP *DELETE* method is used to delete an existing RDF graph at a given IRI.

The HTTP *GET* method is used to retrieve the entire content of an RDF graph identified by the requested IRI. The HTTP *POST* method is used to add the RDF statements in the payload to an existing RDF graph identified by the requested IRI.

If the requested IRI does not point to an existing RDF graph, the 400 `Bad Request` HTTP status code is returned.

The described RESTful protocol complements the SPARQL 1.1 protocol that uses HTTP GET and POST to submit more sophisticated query or update requests to an RDF repository.

8.4.4 SPARQL Service Descriptions

SPARQL helps solving interoperability problems between clients that want to issue queries and servers that should answer them. However, even with SPARQL, a large amount of out-of-band communication is necessary for clients to issue queries. First of all, clients have to know that the SPARQL endpoint exists. Secondly, they have to know if it supports specific SPARQL extensions they may need, for example, aggregates. Finally, they need to know a high-level description of the data available through the endpoint. A standardized and widely deployed solution is needed in order for servers to advertize their capabilities and data availability, and for clients to discover and to understand them.

Four proposals are currently available, but none of them has been widely deployed:

- The SPARQL Service Advertisement and Discovery Language (SADDLE) [53] allows specifying the query language, datatype, inference level, and result format supported by the endpoint. Moreover, it allows specifying which features are not implemented, the named graphs in the repository, and the vocabularies used.
- The DARQ Vocabulary [17, 46] provides a declarative description of the data available from an endpoint, the definition of access patterns limitations, and statistical information about the available data that can be used for query optimization.
- The Vocabulary of Interlinked Datasets (voID) [78] allows one to describe a dataset in terms of statistics of the published data, the topic of the dataset (e.g., Computer Science references), the links to other published datasets, and examples of resources.
- The SPARQL 1.1 Service Description [63] working draft specifies a vocabulary to advertise a SPARQL endpoint in terms of the query language it supports, the SPARQL extensions (e.g., projection functions or support for aggregates) it implements, the entailment regime it adopts, and some statistics (e.g., number of RDF statements) of the RDF datasets that can be queried.

An example of dataset described using voID, Dublic Core, and FOAF vocabularies is presented in [Table 8.51](#).

8.4.5 Entailment Regimes

SPARQL is defined to provide correct answers under different entailment regimes. The first version of SPARQL [45] only supports simple entailment (see Section 2 of [47]). In SPARQL 1.1 Entailment Regimes [59], the semantics of SPARQL queries under RDF and RDFS entailments (see Sections 3 and 4 of [47]) is defined. The working group is currently

specifying other entailment regimes including D-entailment (see Section 5.2 of [47]), OWL 2 Profiles [OWL2Profiles_2009], and RIF [51].

Under simple entailment regimes, a SPARQL engine can find answers only by matching the triple pattern of the query onto the RDF graph of the queried data. Under other entailment regimes, such as RDFS entailment (see Section 4 of [47]), a SPARQL engine can find answers that are not directly specified in the queried graph, but that can be inferred using a set of inference rules.

For instance, the RDF graph in ▶ [Table 8.52](#) describes a simple vocabulary based on RDFS. It asserts that sculptors are artists (line 1), that sculptures are masterpieces (line 2),

■ **Table 8.51**

Example of how DBpedia dataset could be described using using void, Dublic Core, and FOAF vocabularies

```
@prefix void: <http://rdfs.org/ns/void#> .
@prefix ex: <http://example.com/void/datasets/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dcterms: <http://purl.org/dc/terms/> .
ex:DBpedia a void:Dataset ;
    dcterms:title "DBPedia" ;
    dcterms:description "RDF data extracted from Wikipedia" ;
    dcterms:license <http://www.gnu.org/copyleft/fdl.html> ;
    foaf:homepage <http://dbpedia.org/> ;
    void:exampleResource
        <http://dbpedia.org/resource/Berlin> ;
    void:exampleResource
        <http://dbpedia.org/resource/Physics> ;
    void:exampleResource
        <http://dbpedia.org/resource/Ludwig_van_Beethoven> ;
    dcterms:source <http://dbpedia.org/resource/Wikipedia> ;
    dcterms:modified "2008-11-17"^^xsd:date ;
    void:feature :RDFXML ;
    void:sparqlEndpoint <http://dbpedia.org/sparql> .
```

■ **Table 8.52**

An RDF graph that includes a small vocabulary to describe artist and masterpieces as well as two instances

```
1. ex:Sculptor rdfs:subclassOf ex:Artist .
2. ex:Sculpture rdfs:subclassOf ex:MasterPiece .
3. ex:carve rdf:type rdf:Property .
4. ex:carve rdfs:domain ex:Sculptor .
5. ex:carve rdfs:range ex:Sculpture .
6. ex:LaGioconda rdf:type ex:MasterPiece .
7. ex:Rodin ex:carve ex:TheThinker .
```


■ **Table 8.53**

A SPARQL query that can be issued against the RDF graph in [▶ Table 8.49](#)

```
1. SELECT ?x
2. WHERE {
3.   ?x rdf:type ex:MasterPiece .
4. }
```

■ **Table 8.54**

Two of the RDFS entailment rules

Rule Name	If the RDF graph contain	then add
rdfs3	aaa rdfs:range xxx . uuu aaa vvv .	vvv rdf:type xxx .
rdfs9	uuu rdfs:subClassOf xxx . vvv rdf:type uuu .	vvv rdf:type xxx .

that sculptors carve sculptures (from line 3 to line 5), that “La Gioconda” is a masterpiece (line 6), and that Rodin carved “The Thinker” (line 7).

When the query in [▶ Table 8.53](#) is issued against the RDF graph in [▶ Table 8.52](#), a SPARQL engine that operates under a simple entailment checks how the basic graph pattern `?x rdf:type ex:MasterPiece` can be mapped to the queried graph and by instantiating `?x` with `ex:LaGioconda` it generates one match; but it cannot infer that also `ex:TheThinker` is a masterpiece. To do so, the SPARQL engine needs to operate under RDFS entailment.

Under RDFS entailment, a set of rules (see Section 7.3 of [47]) is applied to the explicit RDF statements to derive inferred RDF statements. [▶ Table 8.54](#) reports the two RDFS entailment rules needed to infer the implicit RDF statements that will allow to instance the mapping between `?x` and `ex:TheThinker`.

The rule `rdfs3` can be applied to the triples at line 5 and 7 to derive the RDF statement `ex:TheThinker rdf:type ex:Sculpture`. Then the rule `rdfs9` can be applied to this inferred RDF statement and to the statement at line 2 to derive `ex:TheThinker rdf:type ex:MasterPiece`. Such an RDF statement can finally be used to find that `ex:TheThinker` is also an answer to the query in [▶ Table 8.53](#) under an RDFS entailment regime.

8.4.6 Querying the Entire Semantic Web with SPARQL

SPARQL aims at being the language for querying data published on the Web, but actually it is a query language for RDF either stored natively as RDF or viewed as RDF via middleware. SPARQL alone cannot be used to query the entire Semantic Web. Several solutions have been investigated in order to support the execution of queries over the entire Semantic

Web. Known approaches can be classified in three categories: query federation, data centralization, and dynamic discovery of RDF graphs at query time.

Query federation has been largely investigated in the database area (see [56] for an overview of the concepts developed in this context). The Semantic Web approaches adapt database concepts to RDF graphs published on the Web and can be queried using SPARQL. For instance, both DARQ engine [46] and SemWIQ [33] expose to their users a SPARQL endpoint, but they do not execute the query locally. When they receive a SPARQL query, they transparently decompose it in subqueries, they forward the subqueries to multiple, distributed SPARQL engines, and they eventually compose the results of the subqueries in the result of the original SPARQL query.

Query-answering latency can become an issue in federated query processing; in this case, Semantic Web practitioners normally opt for *data centralization*. The idea is to provide a SPARQL endpoint over a collection of RDF graphs discovered on the Web and replicated in a local repository. For instance, OpenLink replicates in a single site (<http://lod.openlinksw.com/>) the majority of the datasets from the Linking Open Data community project [35] (it includes the datasets DBpedia, DBtunes, and BIO2RDF presented in [♦ Sect. 8.2](#)).

Even with data centralization approaches, queries are executed “only” on selected datasets that are part of the collection. However, the Semantic Web is a single, open, and globally distributed dataspace. Being the Semantic Web open, knowing in advance all data sources, which might be relevant for query answering, is impossible. This openness poses a new challenge that is not addressed by traditional research on federated query processing, and certainly data centralization is only feasible to a limited extent. Embracing this new challenge, Semantic Web researchers are investigating approaches to *dynamically discover RDF graphs* that might be relevant for answering at query execution [29, 34].


Hartig et al. [29] propose an approach to dynamically fetch RDF graphs following RDF links between data sources based on IRIs in the query and in partial results. The RDF graphs pointed by the IRIs are continuously added to the inner dataset. An iterator-based pipeline is used to execute the query over a growing set of potentially relevant data retrieved from the Web. Note that this approach differs from query federation that requires data source to expose a SPARQL endpoint, because it only requires data owners to publish data following the linked data principles [11].

The European Project LarKC developed a solution [LarKC-D5.2.1] similar to the Hartig et al. solution. Instead of following RDF links, it discovers new RDF graphs using Sindice [57]. Sindice, as the other Semantic Web search engines [72, 80], crawls the Semantic Web, indexes discovered RDF graphs, and provides query interfaces to their indexes. The LarKC solution receives a SPARQL query from the user, extracts the basic graph patterns from the WHERE clause, and constructs a set of triple patterns in the form required by Sindice APIs (see [♦ Sect. 8.1.8](#)). At this point the LarKC solution starts an iterative process. At first, it invokes Sindice API and it gets as result, a set of IRIs that points to RDF graphs in which triples conforming to the requested patterns can be found. Then, the LarKC solution loads the remote RDF graph identified by Sindice in the inner dataset and it answers to the user SPARQL query. Sindice provides results in pages; if further

results are available, a link in the current page points to the next results. Iteration after iteration, the LarkC solution fetches an increasing number of pages of results from Sindice, loads an increasing number of remote RDF graphs in the inner dataset, and generates an increasing stream of answers to the user query. The iterative process stops when all the pages of Sindice results are returned and all remote RDF graphs are loaded in the inner data layer.

8.4.7 Continuous SPARQL

Data streams are unbounded sequences of time-varying data elements. They have been recognized in a variety of modern applications, such as network monitoring, traffic engineering, sensor networks, RFID tag applications, telecom call records, and financial applications. Processing of data streams has been largely investigated in the last decade [22], specialized Data Stream Management Systems have been developed, and their features are becoming supported by major database products, such as Oracle and DB2. Thus, the extension of SPARQL for dealing with streaming data is currently investigated.

Streaming SPARQL [68], C-SPARQL (Continuous-SPARQL [14]), and TA-SPARQL (Time-Annotated SPARQL [73]) are proposals for extending SPARQL to stream data management. They introduce the notion of RDF streams as the natural extension of the RDF data model to this scenario, and then extend SPARQL to query RDF streams.  [Table 8.55](#) presents an example of C-SPARQL query that, given a static description of brokers and a stream of financial transactions for all Swiss brokers, computes the amount of their transactions within the last hour.

At line 1, the REGISTER clause is used to tell the C-SPARQL engine that it should register a continuous query, that is, a query that will continuously compute answers to the

[Table 8.55](#)

Example of C-SPARQL, which allows dealing with streams of RDF triples

```

1. REGISTER QUERY TotalAmountPerBroker COMPUTE EVERY 10m AS
2. PREFIX ex: <http://example/>
3. SELECT DISTINCT ?broker ?total
4. FROM <http://brokerscentral.org/brokers.rdf>
5. FROM STREAM <http://stockex.org/market.trdf>
6.   [RANGE 1h STEP 10m]
7. WHERE {
8.   ?broker ex:from ?country .
9.   ?broker ex:does ?tx .
10.  ?tx ex:with ?amount .
11.  FILTER (?country = "CH" )
12. }
13. AGGREGATE { (?total, SUM(?amount), ?broker) }

```

query; the `COMPUTE EVERY` clause states the frequency of every new computation. At line 5, the clause `FROM STREAM` defines the RDF stream of financial transactions, used within the query. Next, line 6 defines the “window” of observation of the RDF stream. Streams, for their very nature, are volatile and for this reason should be consumed on-the-fly; thus, they are observed through a window, including the last elements of the stream, which changes over time. In the example, the window comprises RDF triples produced in the last 1 h, and the window slides every 10 min. The `WHERE` clause is standard; it includes a set of matching patterns and `FILTER` clauses. Finally, at line 13, the `AGGREGATE` function asks the C-SPARQL engine to include in the result set a new variable `?total` that is bound to the sum of the amount of the transaction of each broker.

8.5 Conclusions

This chapter illustrates SPARQL, a language and a protocol to query the Semantic Web. The first part of the chapter progressively introduces the reader to the SPARQL query language by explaining how to:

- Write graph patterns
- Match RDF literals and blank nodes
- Construct constraints using the `FILTER` clause
- Deal with incomplete data using the `OPTIONAL` clause
- Match alternatives using the `UNION` clause
- Use `OPTIONAL` and `FILTER` to implement negation as failure
- Project results using the `SELECT` clause
- Eliminate duplicates using the clauses `DISTINCT` and `REDUCED`
- Order results using the `ORDER BY` clause
- Reference RDF datasets from a SPARQL query using the clauses `FROM`, `FROM NAMED`, and `GRAPH`
- Exchange query request and result over the Web using the SPARQL protocol

All these features allow SPARQL to operate on the *open, decentralized, and fluid* Web. In particular, if the clients know both the RDF vocabulary and the endpoint (the data source location), they can issue `SELECT` and `CONSTRUCT` queries. If they do not know the vocabulary, they can still issue SPARQL queries using the `DESCRIBE` form. If they know multiple endpoints, they can discover which one will be able to answer their queries by using the `ASK` form. A brief description of the semantics of SPARQL closes this first part.

The central part of the chapter illustrates examples of applications, following a historical order. It first describes the bibliographic applications of the “early days” when RDF data were lacking. Then it presents the early adoption of SPARQL in the area of bioscience. Finally, it shows readers a set of SPARQL queries that nowadays can be answered using the interlinked RDF repositories of music-related information and of governmental data.

The final part of the chapter is dedicated to proposed extensions and future trends in SPARQL. In particular, it discusses how the ongoing SPARQL 1.1 standardization effort is

addressing the issues that arose in the community of SPARQL practitioners. Aggregates, subqueries, projections, and negation are presented as the main foreseen extensions to the query language. A RDF graph manipulation language as well as a protocol extension is also discussed. Execution of SPARQL queries under different entailment regimes proposed for standardization is briefly illustrated.

The end of the chapter gives the reader an outlook on the most advanced development in the context of SPARQL, such as vocabularies for describing SPARQL endpoints in order to enable automatic discovery and selection, approaches for querying the entire Semantic Web with SPARQL, and proposals for extending SPARQL to the management of RDF streams.

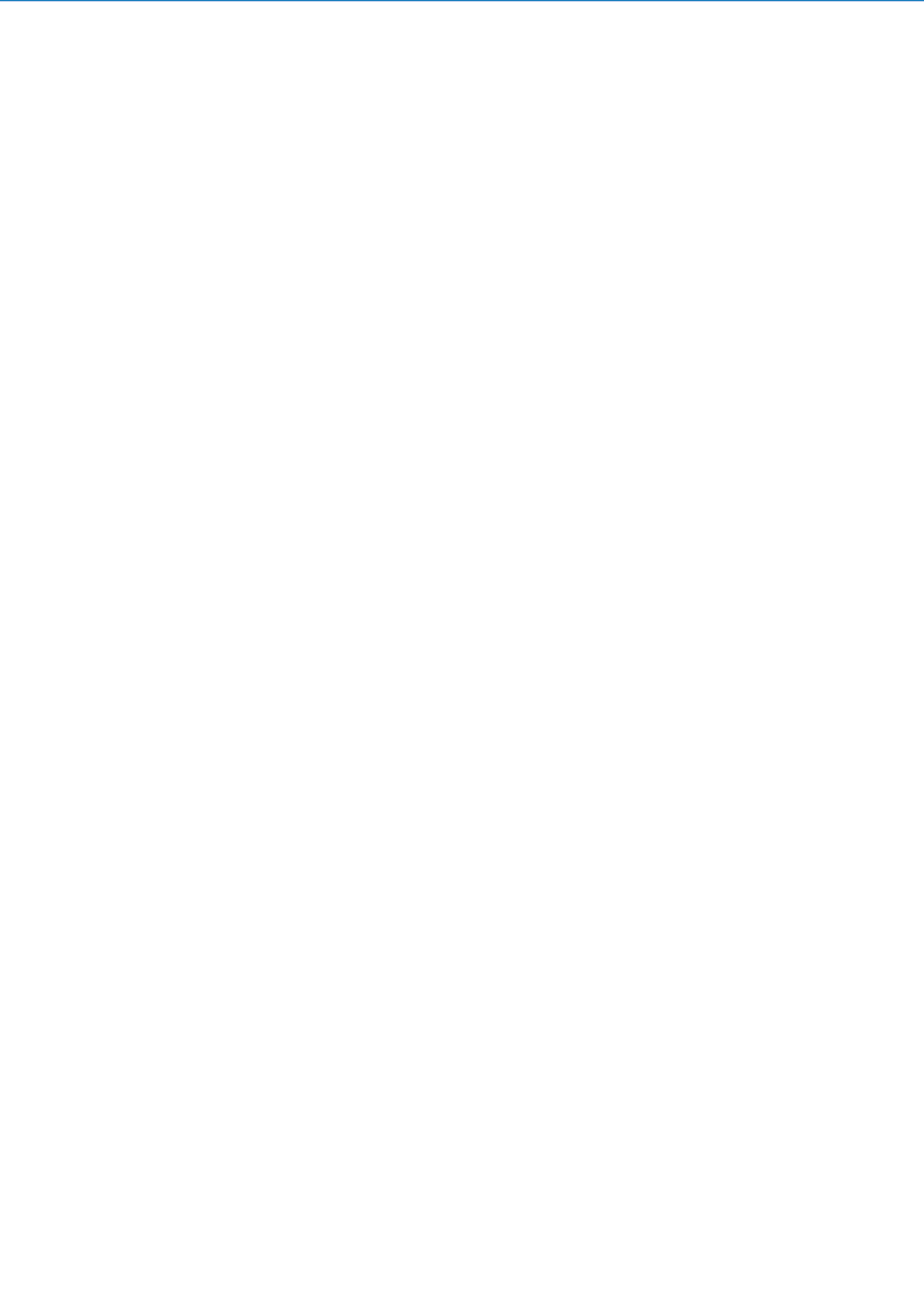
References

1. SPARQL+: ARC2 SPARQL extensions. <http://arc.semsol.org/docs/v2/sparql+>. Accessed 14 Feb 2011
2. ARQ – Documentation and Resources – support for GROUP BY and counting. <http://jena.sourceforge.net/ARQ/group-by.html>. Accessed 14 Feb 2011
3. OpenLink Virtuoso Universal Server: Documentation. Aggregates in SPARQL. <http://docs.openlinksw.com/virtuoso/rdfsparqlaggregate.html>. Accessed 14 Feb 2011
4. Raimond, Y., Sinclair, P., Humfrey, N.J., Smethurst, M.: BBC programmes ontology. <http://purl.org/ontology/po/> (2009). Accessed 14 Feb 2011
5. Beckett, D., Broekstra, J. (ed.): SPARQL query results XML format, W3C Recommendation. World Wide Web Consortium (2008)
6. François, B., Marc-Alexandre, N., Nicole, T., Philippe, R., Jean, M.: Bio2RDF: towards a mashup to build bioinformatics knowledge systems. *J. Biomed. Inform.* **41**(5), 706–716 (2008)
7. Marc-Alexandre, N., Michel, D., Francois, B.: Life science raw data now, W3C-HCLS F2F Meeting, Cambridge. <http://www.slideshare.net/fbelleau/bio2rdf-w3c-hcls2009> (Apr 2009). Accessed 14 Feb 2011
8. Freebase: A list of SPARQL endpoint provided by the Bio2RDF initiative. <http://www.freebase.com/view/user/bio2rdf/public/sparql>. Accessed 14 Feb 2011
9. Stein, L.D.: Integrating biological databases. *Nat. Rev. Genet.* **4**, 337–345 (2003). <http://dx.doi.org/10.1038/nrg1065>
10. Bizer, C., Cyganiak, R., Auer, S., Kobilarov, G.: DBpedia.org – querying wikipedia like a database. In: Proceedings of the 16th International Conference on World Wide Web (WWW 2007), Banff (2007)
11. Christian, B., Tom, H., Tim, B.-L.: Linked data – the story so far. *Int. J. Semant. Web Inf. Syst.* **5**(3), 1–22 (2009)
12. Chinnici, R., Moreau, J.-J., Ryman, A., Weerawarana, S. (ed.): Web Services Description Language (WSDL) version 2.0 part 1: core language, W3C Recommendation. World Wide Web Consortium (2007)
13. Clark, K.G., Feigenbaum, L., Torres, E. (eds.): SPARQL protocol for RDF, W3C Recommendation (2008)
14. Barbieri, D.F., Braga, D., Ceri, S., Della Valle, E., Grossniklaus, M.: C-SPARQL: SPARQL for continuous querying. In: Proceedings of the 18th International World Wide Web Conference (WWW 2009), Madrid (2009)
15. OpenLink Virtuoso Universal Server: Documentation. Standard and user-defined aggregate functions. <http://docs.openlinksw.com/virtuoso/aggregates.html>
16. Cyganiak, R., Bizer, C.: D2R server – publishing relational databases on the web as SPARQL endpoints. In: Proceedings of the 15th International World Wide Web Conference (WWW 2006), Edinburgh (2006)
17. Quilitz, B.: DARQ - federated queries with SPARQL. http://darq.sourceforge.net/index.html#Service_Descriptions (2006)

18. Feigenbaum, L.: DAWG testcases, W3C. <http://www.w3.org/2001/sw/DataAccess/tests/r2> (2004). Accessed 14 Feb 2011
19. D2R Server: DBLP bibliography database. Freie Universität, Berlin. <http://www4.wiwiss.fu-berlin.de/dblp/> (2006). Accessed 14 Feb 2011
20. DBTune: DBTune website: <http://dbtune.org/>. Accessed 14 Feb 2011
21. DCMI Usage Board: DCMI metadata terms. Dublin core metadata initiative. <http://dublincore.org/documents/2006/12/18/dcmi-terms> (2006)
22. Garofalakis, M., Gehrke, J., Rastogi, R.: Data Stream Management: Processing High-Speed Data Streams (Data-Centric Systems and Applications). Springer, Secaucus (2007)
23. Faceted DBLP search engine. <http://dblp.l3s.de/>. Accessed 14 Feb 2011
24. Fielding, R.T., Taylor, R.N.: Principled design of the modern web architecture. *ACM Trans. Internet Technol.* 2(2), 115–150 (2002)
25. Becker, C., Bizer, C.: flickr™ wrappr: precise photo association. <http://www4.wiwiss.fu-berlin.de/flickrwrappr/>. Accessed 14 Feb 2011
26. Brickley, D., Miller, L.: FOAF vocabulary specification. FOAF project. <http://xmlns.com/foaf/spec/20070524.html> (2007)
27. Brickley, D.: WGS84 geo positioning: an RDF vocabulary. <http://www.w3.org/2003/01/geo/> (2004). Accessed 14 Feb 2011
28. guardian.co.uk: Free our data. <http://www.guardian.co.uk/technology/free-our-data>. Accessed 14 Feb 2011
29. Hartig, O., Bizer C., Freytag, J.-C.: Executing SPARQL queries over the web of linked data. In: Proceedings of the Eighth International Semantic Web Conference (ISWC 2009), Chantilly. Lecture Notes in Computer Science, vol. 5823, pp. 293–309. Springer, Heidelberg (2009)
30. Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., Berners-Lee, T.: Hypertext Transfer Protocol – HTTP/1.1 (RFC 2616). In: Internet Engineering Task Force (IETF) (ed.). <http://www.ietf.org/rfc/rfc2616> (1999)
31. Kiryakov, A.: LDSR passes the modigliani test for semantic web. LarKC Blog Post. <http://blog.larkc.eu/?p=1951> (2010)
32. Kjærsmo, K., Passant, A.: SPARQL new features and rationale, W3C Working Draft. <http://www.w3.org/TR/2009/WD-sparql-features-20090702/> (July 2009)
33. Langegger, A., Woß, W., Blochl, M.: A semantic web middleware for virtual data integration on the web. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) Proceedings of the Fifth European Semantic Web Conference (ESWC 2008), Tenerife. Lecture Notes in Computer Science, vol. 5021, pp. 493–507. Springer, Heidelberg (2008)
34. Gallizo, G., Kerrigan, M., Bradesko, L., Fortuna, B.: D5.2.1 rapid prototype of the LarKC. http://www.larkc.eu/wp-content/uploads/2008/01/larkc_d521_rapid-prototype-of-the-larkc.pdf (2009)
35. Linking open data community project. <http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>. Accessed 14 Feb 2011
36. MacManus, R.: Modigliani test: the semantic web's tipping point. ReadWriteWeb post. http://www.readwriteweb.com/archives/the_modigliani_test_semantic_web_tipping_point.php (2010)
37. Giasson, F., Raimond, Y.: Music ontology specification. <http://musicontology.com/> (2007)
38. Nucleic Acids Res. 37, Web server issue. http://nar.oxfordjournals.org/content/vol37/suppl_2/index.dtl (2009)
39. ARQ – Documentation and Resources - support for negation. <http://jena.sourceforge.net/ARQ/negation.html>. Accessed 14 Feb 2011
40. Smith, M.K., Welty, C., McGuinness, D.L.: OWL web ontology language – guide, W3C Recommendation. <http://www.w3.org/TR/owl-guide/> (Feb 2004)
41. Motik, B., Grau, B.C., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C.: OWL 2 web ontology language profiles, W3C Recommendation. <http://www.w3.org/TR/owl2-profiles/> (Oct 2007)
42. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and complexity of SPARQL. In: Cruz, I. F., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) Proceedings of the Fifth International Semantic Web Conference (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 30–43. Springer, Heidelberg (2006)
43. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and complexity of SPARQL. *ACM Trans. Database Syst.* 34(3) (2009)
44. Prud'hommeaux, E., Seaborne, A. (ed.): SPARQL query language for RDF, W3C Working Draft.

- <http://www.w3.org/TR/2006/WD-rdf-sparql-query-20061004/> (Oct 2006)
45. Prud'hommeaux, E., Seaborne, A. (ed.): SPARQL query language for RDF, W3C Recommendation. World Wide Web Consortium (2008)
 46. Quilitz, B., Leser, U.: Querying distributed RDF data sources with SPARQL. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) Proceedings of the Fifth European Semantic Web Conference (ESWC 2008), Tenerife. Lecture Notes in Computer Science, vol. 5021, pp. 524–538. Springer, Heidelberg (2008)
 47. Hayes, P.: RDF semantics, W3C Recommendation. <http://www.w3.org/TR/rdf-mt/> (Feb 2004)
 48. RDF: Query Language and code. <http://search.cpan.org/dist/RDF-Query/>. Accessed 14 Feb 2011
 49. Duerst, M., Suignard, M.: Internationalized Resource Identifiers (IRIs) (RFC 3987). In: Internet Engineering Task Force (IETF) (ed.). <http://www.ietf.org/rfc/rfc3987> (2005)
 50. Phillips, A., Davis, M.: RFC 4647 matching of language tags (2006)
 51. Kifer, M., Boley, H.: RIF overview, W3C Working Draft. <http://www.w3.org/TR/rif-overview/> (Oct 2009)
 52. Glaser, H., Millard, I.C.: RKB explorer: application and infrastructure. In: Proceedings of the Semantic Web Challenge 2007 co-located with ISWC 2007 + ASWC 2007, Busan (2007)
 53. Clark, K.: SPARQL Service Advertisement and Discovery Language (SADDLE). <http://www.w3.org/2001/sw/DataAccess/proto-wd/saddle.html> (2005)
 54. Aduna, B.V.: The SeRQL query language (revision 3.0). <http://www.openrdf.org/doc/sesame2/users/ch09.html> (2002–2008)
 55. The Semantic Web Conference Corpus a.k.a. the Semantic Web Dog Food Corpus. <http://data.semanticweb.org/>. Accessed 14 Feb 2011
 56. Sheth, A.P., Larson, J.A.: Federated database systems for managing distributed, heterogeneous, and autonomous databases. ACM Comput. Surv. 22(3), 183–236 (1990)
 57. Oren, E., Delbru, R., Catasta, M., Cyganiak, R., Stenzhorn, H., Tummarello, G.: Sindice.com: a document-oriented lookup index for open linked data. Int. J. Metadata Semant. Ontol. 3(1), 37–52 (2008)
 58. Miles, A., Bechhofer, S.: SKOS Simple Knowledge Organization System, W3C Proposed Recommendation. <http://www.w3.org/TR/2009/PR-skos-reference-20090615/> (June 2009)
 59. Glimm, B., Parsia, B.: SPARQL 1.1 entailment regimes, W3C Working Draft. <http://www.w3.org/TR/2009/WD-sparql11-entailment-20091022/> (Oct 2009)
 60. Ogbuji, C.: Uniform HTTP protocol for managing RDF graphs, W3C Working Draft. <http://www.w3.org/TR/2009/WD-sparql11-http-rdf-update-20091022/> (Oct 2009)
 61. Johnston, S., Feigenbaum, L., SPARQL 1.1 protocol for RDF, W3C Working Draft. <http://www.w3.org/TR/2009/WD-sparql11-protocol-20091022/> (Oct 2009)
 62. Harris, S., Seaborne, A., SPARQL 1.1 query, W3C Working Draft. <http://www.w3.org/TR/2009/WD-sparql11-query-20091022/> (Oct 2009)
 63. Williams, G.T.: SPARQL 1.1 service description, W3C Working Draft. <http://www.w3.org/TR/2009/WD-sparql11-service-description-20091022/> (Oct 2009)
 64. Schenck, S., Gearon, P.: SPARQL 1.1 update, W3C Working Draft. <http://www.w3.org/TR/2009/WD-sparql11-update-20091022/> (Oct 2009)
 65. SPARQL implementation coverage report. <http://www.w3.org/2001/sw/DataAccess/tests/implementations> (2008). Accessed 14 Feb 2011
 66. Schmidt, M.: Foundations of SPARQL query optimization. Ph.D. thesis, Albert-Ludwigs-Universität Freiburg (Germany). http://www.informatik.uni-freiburg.de/~mschmidt/docs/diss_final01122010.pdf. Accessed 14 Feb 2011
 67. Seaborne, A., Manjunath, G., Bizer, C., Breslin, J., Das, S., Davis, I., Harris, S., Idehen, K., Corby, O., Kjernsmo, K., Nowack, B.: SPARQL update – a language for updating RDF graphs, W3C Member Submission. <http://www.w3.org/Submission/2008/SUBM-SPARQL-Update-20080715/> (July 2008)
 68. Bolles, A., Grawunder, M., Jacobi, J.: Streaming SPARQL – extending SPARQL to process data streams. In: Proceedings of the Fifth European Semantic Web Conference (ESWC 2008), Tenerife. Lecture Notes in Computer Science, vol. 5021, pp. 448–462. Springer, Heidelberg (2008)
 69. ARQ - Documentation and Resources – support for nested SELECTs. <http://jena.sourceforge.net/ARQ/sub-select.html>. Accessed 14 Feb 2011

70. Erling, O.: SPARQL extensions for subqueries. <http://www.openlinksw.com/weblog/oerling/?id=1296>. Accessed 14 Feb 2011
71. Suchanek, F.M., Kasneci, G., Weikum, G.: YAGO: a large ontology from wikipedia and WordNet. *J. Web Semant.* **6**(3), 203–217. (2008)
72. Ding, L., Finin, T.W., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V., Sachs, J.: Swoogle: a search and metadata engine for the semantic web. In: Proceedings of the 13th ACM Conference on Information and Knowledge Management (CIKM 2004), Washington, DC, pp. 652–659. ACM, New York (2004)
73. Rodríguez, A., McGrath, R., Liu, Y., Myers, J.: Semantic management of streaming data. In: Proceedings of the Second International Workshop on Semantic Sensor Networks (SSN 2009), Collocated with the Eighth International Semantic Web Conference (ISWC 2009), Chantilly
74. Berners-Lee, T.: Putting government data online. <http://www.w3.org/DesignIssues/GovData.html> (2009)
75. Beckett, D., Berners-Lee, T.: Turtle - terse RDF triple language, W3C Team Submission. <http://www.w3.org/TeamSubmission/turtle/> (Jan 2008)
76. UK Cabinet Office. Pioneer of the world wide web to advise the government on using data. http://www.cabinetoffice.gov.uk/newsroom/news_releases/2009/090610_web.aspx (2009)
77. Obama, B.: Transparency and Open Government, memorandum for the heads of executive departments and agencies. http://www.whitehouse.gov/the_press_office/Transparency_and_Open_Government/ (2009)
78. Alexander, K., Cyganiak, R., Hausenblas, M., Zhao, J.: Vocabulary of Interlinked Datasets (void). <http://semanticweb.org/wiki/VoID> (2009)
79. RDF Data Access Working Group. <http://www.w3.org/2001/sw/DataAccess/>. Accessed 14 Feb 2011
80. d'Aquin, M., Motta, E., Sabou, M., Angeletou, S., Gridinoc, L., Lopez, V., Guidi, D.: Toward a new generation of semantic web applications. *IEEE Intell. Syst.* **23**(3), 20–28 (2008)
81. Malhotra, A., Melton, J., Walsh, N. (eds): XQuery 1.0 and XPath 2.0 functions and operators, W3C Recommendation. World Wide Web Consortium. <http://www.w3.org/TR/2007/REC-xpath-functions-20070123/> (Jan 2007)
82. Biron, P.V., Malhotra, A. (eds.): XML schema part 2: datatypes second edition, W3C Recommendation. <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/>. Latest version <http://www.w3.org/TR/xmlschema-2/> (Oct 2004)



9 KR and Reasoning on the Semantic Web: OWL

Ian Horrocks¹ · Peter F. Patel-Schneider²

¹Oxford University Computing Laboratory, Oxford, UK

²Alcatel-Lucent Bell Labs Research, New Jersey, USA

9.1	<i>Introduction</i>	366
9.2	<i>History and Influences</i>	367
9.2.1	Description Logics	367
9.2.2	OIL and DAML+OIL	369
9.2.3	OWL and OWL 2	371
9.3	<i>The OWL 2 Language</i>	373
9.3.1	OWL 2 Ontologies	374
9.3.2	OWL 2 Datatypes	374
9.3.3	OWL 2 Entities	375
9.3.4	Expressions	376
9.3.5	Axioms	378
9.3.6	Annotations	381
9.3.7	Global Restrictions	382
9.4	<i>Semantics for OWL 2</i>	384
9.4.1	OWL 2 RDF-Based Semantics	384
9.5	<i>OWL 2 Profiles</i>	385
9.5.1	OWL 2 EL	385
9.5.2	OWL 2 QL	386
9.5.3	OWL 2 RL	387
9.6	<i>OWL Tools and Applications</i>	388
9.6.1	Ontology Design Tools	389
9.6.2	Reasoning in Deployed Applications	391
9.7	<i>Summary</i>	393
9.8	<i>Cross-References</i>	394

Abstract: OWL is the ontology language recommended by the W3C. OWL is heavily based on the knowledge representation languages called Description Logic, which provide the basic representation features of OWL. OWL also includes facilities that integrate it into the mainstream of the Web, including use of Internationalized Resource Identifiers (IRIs) as names, XML Schema datatypes, and ontologies as Web documents, which can then import other OWL ontologies over the Web. Because OWL is based on Description Logics, its constructs have a well-denned meaning and there are tools that effectively perform inference within OWL, enabling the discovery of information that is not explicitly stated in OWL ontologies.

9.1 Introduction

A major feature of the Semantic Web is the ability to provide definitions for objects and types of objects that are accessible and manipulable from within the Semantic Web. In Computer Science, a collection of these sorts of definitions about a particular domain is called an ontology, although philosophers may (and probably will) have a different understanding of what constitutes an ontology.

It is this ability to provide machine-processable definitions and meanings that most divides the Semantic Web from the Visual Web of HTML, CSS, SVG, etc., which is focused on presenting information to humans, and the Syntactic Web of XML, which is focused on the transfer of uninterpreted data between applications (which themselves are written by humans who have extra-Web knowledge of what the data mean).

OWL [30,62] is an ontology language designed for use in the Semantic Web and is the language recommended by W3C for this use. OWL has influences from quite a number of sources, but its main representational facilities are directly based on Description Logics [1]. This basis confers upon OWL a logical framework, including both syntax and model-theoretic semantics. OWL also inherits from Description Logics a concern for practical reasoning and effective, readily available reasoners, for example, the Description Logic reasoners Pellet [74] and HermiT [51], both of which have been extended to handle all of OWL.

OWL is also completely integrated into the Semantic Web and uses other W3C recommendations. The official transfer syntax for OWL ontologies is RDF/XML, a syntax for transferring RDF graphs. Names (of individuals, etc.) in the Semantic Web are Internationalized Resource Identifiers (IRIs) [66], so names in OWL are IRIs. OWL uses XML Schema datatypes [9] to define and type the data values that it uses. OWL ontologies are Web documents and are stored and retrieved just as other Web documents. OWL includes constructs for identifying ontologies and importing one ontology into another. OWL ontologies can also be combined with rules using the new W3C Rule Interchange Format (RIF) standard [67].

The remainder of this chapter will discuss the history of OWL, and the various influences that have shaped it; describe the results of these influences, namely the OWL language; and explain how OWL is used in applications, and in particular how OWL

reasoning services are used in practice. The focus will be on OWL 2 [56], an extension and revision of OWL that became a W3C Recommendation on October 27, 2009.

9.2 History and Influences

9.2.1 Description Logics

Because OWL is heavily based on Description Logics, it inherits their history and shares their influences.

The history of Description Logics started with attempts to formalize Semantic Nets, which themselves were attempts to provide a sort of natural representation based on labeled graphs (nets). A major problem with Semantic Nets was that there was no formal meaning for their graphs, leading to conflicts over just what complex nets meant when divorced from a particular system that provided some data manipulation facilities for Semantic Nets. Another early influence on Description Logics were frame systems, such as FRL [68], which had many of the characteristics of Semantic Nets, but grouped related information together into a frame.

The representation language KL-ONE [10,11] was a knowledge representation system that had many of the characteristics of Semantic Nets and frame systems. Not long after its inception, there were attempts to provide a full formal semantics for KL-ONE [40]. KL-ONE with this semantics can be considered as the first proto-Description Logic.

It was then quickly determined that the systems that manipulated KL-ONE constructs were incomplete with respect to the semantics, that is, they systematically were unable to make conclusions that were implicit in the information given to them; shortly thereafter it was determined that *complete* inference for KL-ONE was very difficult, in fact undecidable [72], and that inference even in very limited subsets of KL-ONE was worst-case intractable [40]. This led to the development of simpler representation systems, like KANDOR [61]. The aim for KANDOR was to design a useful representation language in which complete inference would be easy (both easy to implement and easy to decide). Unfortunately, the designers of KANDOR were able to achieve easy complete inference only by imposing severe limits on its expressive power.

The Description Logic community then split into three groups. One group continued work on simple Description Logics with complete or nearly complete but worst-case-tractable reasoning algorithms, such as the one implemented in the CLASSIC system [63]. Another group continued work on more expressive Description Logics that had worst-case intractable reasoning, such as those implemented in the KRIS and CRACK systems [3,12]. This group was more interested in the formal aspects of these Description Logics. A third group built partial reasoners for very expressive, that is, undecidable, Description Logics, such as the one implemented in the LOOM system [45].

Eventually, with the FaCT system [27], it was shown that it was possible to provide a reasoner for an expressive Description Logic (*SH*) that, in spite of the worst-case intractability of basic reasoning problems, nonetheless provided complete and effective

reasoning in most cases. This invigorated the community, as now Description Logics that contained useful representational constructs could be provided with effective reasoners and thus could be used in applications.

In spite of this success, continuing concerns about tractability, particularly in applications requiring very large ontologies or datasets, have rekindled interest in smaller Description Logics where reasoning is worst-case tractable. Research in this area has focused on two main families of Description Logics: the \mathcal{EL} family and the DL-Lite family. The advantage of these logics is that standard reasoning problems have either PTime complexity (for members of the \mathcal{EL} family) or are in AC^0 (for members of the DL-Lite family).

SR₀IQ(D) Modern Description Logics are generally quite alike, with most using the same syntax for their constructs and the same first-order semantics. Even the names of many of the modern Description Logics are rather similar as they are built up from letters that signify features of the logic.

The Description Logics that underlie the various Semantic Web ontology languages are all extensions to the well-known Description Logic \mathcal{ALC} [73] plus transitively closed primitive roles [70]. This Description Logic was called \mathcal{S} due to its relationship with the propositional (multi) modal logic $\mathbf{S4}_m$ [71] (it can also be called \mathcal{ALC}_{R^+} , but this name is cumbersome to add letters to representing additional features). \mathcal{S} is then extended to include features such as role inclusion axioms (\mathcal{H} , or \mathcal{R} if role chains are allowed), nominals (\mathcal{O}), inverse roles (\mathcal{I}), and number restrictions (\mathcal{Q} if qualified, \mathcal{N} if not).


The syntax and semantics of these features is summarized in [Fig. 9.1](#), where A is a concept name, C and D are concepts, R and S are roles, \mathbf{R}_+ is the set of transitive roles, o is

Construct Name	Syntax	Semantics		
atomic concept	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$	\mathcal{S}	
atomic role	R	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$		
transitive role	$R \in \mathbf{R}_+$	$R^{\mathcal{I}} = (R^{\mathcal{I}})^+$		
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$		
disjunction	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$		
negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$		
exists restriction	$\exists R.C$	$\{x \mid \exists y. \langle x, y \rangle \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}$		
value restriction	$\forall R.C$	$\{x \mid \forall y. \langle x, y \rangle \in R^{\mathcal{I}} \text{ implies } y \in C^{\mathcal{I}}\}$		
role hierarchy	$R \sqsubseteq S$	$R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$		\mathcal{H}
role chains	$R_1 \circ \dots \circ R_n \sqsubseteq S$	$R_1^{\mathcal{I}} \circ \dots \circ R_n^{\mathcal{I}} \subseteq S^{\mathcal{I}}$		\mathcal{R}
nominal	$\{o\}$	$\{o^{\mathcal{I}}\}$	\mathcal{O}	
inverse role	R^-	$\{\langle x, y \rangle \mid \langle y, x \rangle \in R^{\mathcal{I}}\}$	\mathcal{I}	
number restriction	$\geq nP$	$\{x \mid \#\{y. \langle x, y \rangle \in P^{\mathcal{I}}\} \geq n\}$	\mathcal{N}	
	$\leq nP$	$\{x \mid \#\{y. \langle x, y \rangle \in P^{\mathcal{I}}\} \leq n\}$		
qualified number restriction	$\geq nP.C$	$\{x \mid \#\{y. \langle x, y \rangle \in P^{\mathcal{I}} \text{ and } \in C^{\mathcal{I}}\} \geq n\}$	\mathcal{Q}	
	$\leq nP.C$	$\{x \mid \#\{y. \langle x, y \rangle \in P^{\mathcal{I}} \text{ and } \in C^{\mathcal{I}}\} \leq n\}$		

Fig. 9.1

Syntax and semantics of the \mathcal{S} family of Description Logics

an individual name, P is a simple role (i.e., one that is not transitive and has no transitive sub-role), and n is a nonnegative integer. Note that, in order to retain decidability, it is necessary to impose some restrictions on the property hierarchy and on the use of roles in number restrictions; for example, transitive roles cannot be used in number restrictions. A complete description of these languages and the relevant restrictions on roles can be found in the Description Logic literature [29,32,33].

These logics can also be extended with *concrete domains*, which allow for the use of “concrete” types, such as the integers, values, such as the integer “5,” and predicates such as integer comparisons [2,41,43]. A simplified form of concrete domains, known as Datatypes, is denoted by appending (**D**) to the name of the logic, for example, $SHOIN(\mathbf{D})$ [31]; Datatypes restrict the interactions between concrete and “abstract” parts of a knowledge base so as to avoid problems of undecidability and to simplify implementation, and are widely used in ontology languages, including OWL and OWL 2 [49]. The syntax and semantics of Datatypes is summarized in  Fig. 9.2, where D is a datatype name, T is a concrete role, v is a data value, and n is a nonnegative integer.

9.2.2 OIL and DAML+OIL

The first significant effort to build a language combining Description Logics and the Semantic Web was OIL (the Ontology Inference Layer) [28], which was developed within the On-To-Knowledge research project (see <http://www.ontoknowledge.org/>) funded by the European Union. The OIL language was explicitly designed as “a web-based representation and inference language for ontologies [that combines] the widely used modeling primitives from frame-based languages with the formal semantics and reasoning services provided by description logics” (see <http://www.ontoknowledge.org/oil/>). OIL was so closely tied to Description Logics that the semantics was specified as a mapping from its syntax to the Description Logic $SHIQ$ [17,33]. From this mapping, OIL gained both

Construct Name	Syntax	Semantics
datatype	D	$D^{\mathcal{I}} \subseteq \Delta_D^{\mathcal{I}}$
data value	v	$v^{\mathcal{I}} \in \Delta_D^{\mathcal{I}}$
concrete role	T	$T^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta_D^{\mathcal{I}}$
enumerated datatype	$\{v_1, \dots, v_n\}$	$\{v_1^{\mathcal{I}}, \dots, v_n^{\mathcal{I}}\}$
exists restriction	$\exists T.D$	$\{x \mid \exists y.\langle x, y \rangle \in T^{\mathcal{I}} \text{ and } y \in D^{\mathcal{I}}\}$
value restriction	$\forall T.D$	$\{x \mid \forall y.\langle x, y \rangle \in T^{\mathcal{I}} \text{ implies } y \in D^{\mathcal{I}}\}$
number	$\geq nT$	$\{x \mid \#\{y.\langle x, y \rangle \in T^{\mathcal{I}}\} \geq n\}$
restriction	$\leq nT$	$\{x \mid \#\{y.\langle x, y \rangle \in T^{\mathcal{I}}\} \leq n\}$
qualified number	$\geq nT.D$	$\{x \mid \#\{y.\langle x, y \rangle \in T^{\mathcal{I}} \text{ and } y \in D^{\mathcal{I}}\} \geq n\}$
restriction	$\leq nT.D$	$\{x \mid \#\{y.\langle x, y \rangle \in T^{\mathcal{I}} \text{ and } y \in D^{\mathcal{I}}\} \leq n\}$

 Fig. 9.2

Syntax and semantics of Description Logic Datatypes

a clear semantics and a means to exploit the reasoning services of Description Logic systems such as FaCT [27], RACER [22], and Pellet [64] that implement (most of) *SHIQ(D)*. OIL was also influenced by frame systems as its syntax provided for the grouping of constructs related to a particular individual or class, thus providing the appearance of a single construct per individual or class.

There were three syntaxes for OIL: a text-based syntax, an XML syntax, and a syntax based on RDF (Resource Description Framework – see <http://www.w3.org/RDF/>). The first of these syntaxes was meant for presentation use only, allowing OIL ontologies to be easily viewed without the verbose additions needed in the other two syntaxes. The XML and RDF syntaxes were suitable for transferring OIL ontologies in the Semantic Web, the latter allowing OIL ontologies to be written directly as RDF graphs. OIL even used the same set of names as RDF (i.e., `rdfs:Class` and `rdfs:subClassOf`), permitting non-OIL RDF to be used along with OIL ontologies. So in OIL there was, for the first time, the adaptation of Description Logic constructs for the Semantic Web. OIL, however, did not have complete integration with RDF. Some of the Description Logic constructs in OIL, particularly `exists`, `value`, and number restrictions, were not RDF classes but instead had a frame-like syntax, and were not RDF classes.

The DARPA Agent Markup Language (DAML) program then joined into the above effort to update and modify OIL to provide an even closer relationship to the Semantic Web, particularly RDFS (RDF Schema – see <http://www.w3.org/TR/rdf-schema/>). The result of this collaboration was DAML+OIL (see <http://www.w3.org/Submission/2001/12/>).

From the point of view of language constructs, the differences between OIL and DAML+OIL are relatively trivial. Although there is some difference in “keyword” vocabulary, there is usually a one-to-one mapping of constructors, and in the cases where the constructors are not completely equivalent, simple translations are possible. However, the frame structure of OIL’s syntax is much less evident in DAML+OIL, with the result that a DAML+OIL ontology is more Description-Logic-like in that it consists largely of a relatively unstructured collection of subsumption and equality axioms. This can make it more difficult to use DAML+OIL with frame-based tools such as PROTÉGÉ [21] or OILED [7], because the axioms may be susceptible to many different frame-like groupings [6].

The initial release of DAML+OIL did not include any specification of datatypes. The language was, however, subsequently extended with arbitrary datatypes from the XML Schema type system [9], which can be used in restrictions (slot constraints) and range constraints. As in *SHOQ(D)* [31], a clean separation is maintained between instances of “object” classes (defined using the ontology language) and instances of datatypes (defined using the XML Schema type system). In particular, it is assumed that the domain of interpretation of object classes is disjoint from the domain of interpretation of datatypes, so that an instance of an object class (e.g., the individual *Italy*) can never have the same interpretation as a value of a datatype (e.g., the integer 5), and that the set of object properties (which relate individuals to individuals) is disjoint from the set of datatype properties (which relate individuals to datatype values).

9.2.3 OWL and OWL 2

DAML+OIL did solve the problem of the syntactic disconnect with RDFS; there was, however, no semantic integration between DAML+OIL and RDF – in fact RDF did not yet have a formal semantics. Further, DAML+OIL was not recommended by any organization beyond the two projects that had sponsored its development.

Therefore, some of the people involved in the design of DAML+OIL decided to try to create a W3C recommendation for an ontology language to be tightly integrated into the Semantic Web. To this end, DAML+OIL was submitted to W3C (see <http://www.w3.org/Submission/2001/12/>), and a Web Ontology Working Group (WebOnt) was subsequently chartered to produce a Semantic Web Ontology Language based on DAML+OIL, but more tightly integrated with RDF and RDFS as well as other Web standards. At the same time, the W3C RDF Core Working Group was, among other things, developing a formal semantics for RDF and RDFS [24].

The result of WebOnt was OWL, the W3C Web Ontology Language (see <http://www.w3.org/2004/OWL/>). OWL had an official Semantic Web syntax in the form of RDF graphs, and could thus be transferred and manipulated using Semantic Web tools. There was also an internal syntax for OWL, which was easier to manipulate formally, and there soon came to be a compact frame-like syntax for OWL called the Manchester Syntax [26]. OWL had a model-theoretic semantics compatible with the semantics of Description Logics, so Description Logics reasoners could, in principle, provide reasoning facilities for OWL as they did for OIL and DAML+OIL. However, the Description Logic *SHOIN(D)* on which OWL was based was more expressive than those supported by existing reasoners, and it was not until more than a year after the completion of OWL that suitable reasoning algorithms and implementations became available.

OWL used the vocabulary of RDF and RDFS where possible, so RDF and RDFS tools could process OWL ontologies that fit into their limited expressive power, as they could for DAML+OIL. OWL's model-theoretic semantics was fine-tuned to be as compatible as possible with the new semantics for RDF and RDFS so that the constructs common with RDF and RDFS had compatible meaning. OWL included mechanisms to import other ontologies and Semantic Web documents across the Semantic Web; it also included a way of adding extralogical information into ontologies to be used, for example, to record the status of particular classes or individuals.

To obtain the desirable implementation characteristics of Description Logics in OWL, it was necessary to limit the ways in which some constructs of RDF and RDFS were used. This required, in particular, restricting the use of RDF and RDFS syntax to that which corresponded to syntactically coherent OWL, ruling out both malformed OWL syntax and the use of RDF, RDFS, and OWL vocabulary as individual, class, or property names in an OWL ontology. It additionally required vocabulary separation, that is, ensuring that the sets of names used for classes, properties, and individuals were disjoint. Ontologies satisfying these restrictions were guaranteed to have the desirable characteristics of Description Logics, and were called OWL DL ontologies. Ontologies that violated these

restrictions were called OWL Full ontologies. Description Logic reasoners would not be complete for OWL Full ontologies, and, indeed, reasoning in OWL Full is trivially undecidable (see [30] for a detailed discussion). The advantage of OWL Full was that any RDF or RDFS document could be used as an OWL Full ontology.

Although very successful, OWL did not, of course, satisfy all user requirements. After extensive discussions between users, theoreticians, and implementers, in particular at the 2005 OWL Experiences and Directions Workshop (see <http://www.mindswap.org/2005/OWLWorkshop/>), it was decided to address some of these requirements via an incremental revision of OWL, provisionally called OWL 1.1. The initial goal of OWL 1.1 was to exploit recent developments in Description Logic research in order to address some of the expressivity limitations of the OWL. However, as the design of OWL 1.1 progressed, it was decided to also address performance requirements by exploiting research into smaller Description Logics with desirable computational properties.

The development of OWL 1.1 was initially undertaken by an informal group of language users and developers. After the original specification reached a mature state, and first implementations were released, the OWL 1.1 proposal was submitted to W3C as a Member Submission (see <http://www.w3.org/Submission/2006/10/>); this was then taken as a starting point for a new W3C Working Group that was officially formed in September 2007. As work on the new language progressed, the initial Member Submission evolved significantly, and the Working Group eventually decided to call the new language OWL 2 to indicate a substantial step in the evolution of the language.

OWL 2 is based on $SR\mathcal{OIQ}(\mathcal{D})$ and so extends OWL with qualified cardinality restrictions and with significantly extended expressivity with respect to properties, for example, the ability to assert that properties are reflexive, irreflexive, asymmetric, and disjoint, and the ability to compose properties into property chains. OWL 2 also weakens the name separation restriction imposed in OWL – in OWL 2 the same name can be used for a class, a property, and an individual, a feature known as punning. In addition, OWL 2 provides greatly extended support for datatypes, including many of the XML Schema datatypes and facets, and for annotations, including, for example, the ability to annotate axioms as well as entities. Finally, OWL 2 also provides a limited form of database style keys.

In addition to increasing the expressive power of the complete language, OWL 2 also defines several *profiles*: language fragments that have desirable computational properties (see <http://www.w3.org/TR/owl2-profiles/>). These include a profile based on DL Lite, a Description Logic for which standard reasoning problems can be reduced to SQL query answering; a profile based on \mathcal{EL}^{++} , a Description Logic for which standard reasoning problems can be performed in polynomial time; and a profile based on Description Logic Programs (DLP) [20], a logic for which query answering can be implemented using rule-based techniques that have been shown to scale well in practice. The old OWL Lite profile was deprecated as it provides no significant computational advantage with respect to OWL.

In addition to language features, OWL 2 boasts several “convenience” features, including an improved specification in both BNF and UML formats, a fully validated

XML Syntax and the use of Manchester syntax as a text-based and more human-friendly syntax.

9.3 The OWL 2 Language

Because OWL is heavily based on Description Logics, OWL 2 exhibits many characteristics of typical Description Logics. In particular, it describes the domain in terms of individuals, classes (called *concepts* in Description Logics), properties (called *roles* in Description Logics), and datatypes and values (called *concrete domains* in Description Logics). Individual names, for example, “John,” refer to elements of the domain; concepts, for example, “University,” describe sets of individuals having similar characteristics; roles, for example, “studiesAt,” describe relationships between pairs of individuals (such as “John studiesAt Oxford”); and datatypes, for example, “integer,” describe sets of data values, for example, “18.” Class descriptions can also be composed from all of the above components using various constructors, including, for example, the Booleans.

Like a Description Logic knowledge base, an OWL 2 ontology consists of a set of axioms and facts that describe the domain, for example, asserting that GradStudent is a subclass of Student, that John is a Student or that John has Age 18. Finally, like a Description Logic, OWL 2 can be seen as a fragment of first-order logic (FOL), and is given a formal semantics based on first-order model theory (although it could equally well be given via a translation into Description Logic, or even directly into FOL).

Because OWL 2 is an ontology language for the Semantic Web it has some differences from most Description Logics and does some things in different ways from Description Logics.

These differences start with the names used in OWL 2, which are IRIs, the names that underlie the Semantic Web (and indeed the Web itself) [66]. As IRIs tend to be very long, OWL 2 syntaxes provide facilities for short forms of names, roughly the same as QNames used by SPARQL [77]. So, for example, OWL 2 syntaxes allow `owl:Thing` as a short form of <http://www.w3.org/2002/07/owl#Thing>. OWL 2 also allows anonymous individuals (individuals without global names), written out as in the RDF syntax for blank nodes, for example, `_:id`. OWL 2 does not assume that different names refer to different entities, so, for example, `ex:Jack` and `ex:John` can both be names for the same person (this is discussed in more detail in [▶ Sect. 9.3.5](#)); nor does it assume that the names used for individuals, classes, and properties are disjoint (as was the case in OWL), so, for example, the same name could be used to denote both a class and an individual (this is discussed in more detail in [▶ Sect. 9.3.3](#)).

Moreover, OWL 2 largely uses the datatype facilities of XML Schema [9], including floating point numbers, instead of the more mathematical (and easier to work with) datatypes common to most Description Logics. The set of supported datatypes and facets is defined in the OWL 2 datatype map, which will be discussed in more detail in [▶ Sect. 9.3.2](#).

OWL 2 has several syntaxes. The standard syntax of the Semantic Web, RDF/XML [8], is the one syntax that all OWL 2 implementations must support. However, as RDF/XML is very verbose and very hard to read, there are other syntaxes for OWL 2, including an XML syntax for integration with XML tools, a functional-style syntax [59] that is used for precision and in formal documents, and an easier to read syntax designed for presentation, called the Manchester syntax [55]. Manchester syntax will be used in the remainder of this chapter, precisely because it is designed for presentation to humans.

9.3.1 OWL 2 Ontologies

OWL 2 has the notion of an ontology – meant to be a collection of related information that describes a domain. These ontologies can (and generally are) stored as Web documents and can be combined into larger collections of information. In contrast to a Description Logic knowledge base, where conceptual and instance level statements are usually separated into, respectively, a set of Tbox axioms and a set of Abox assertions, an OWL 2 ontology consists of a single set of axioms that includes both conceptual and instance level statements. OWL is nonstandard in this regard: ontologies are more typically thought of as describing only the structure of a domain (in terms of classes and properties), and not as describing a particular situation (in terms of instances of classes and properties); in this more common usage, an ontology is therefore equivalent to a Description Logic Tbox, and not to the combination of a Tbox and an Abox. In addition to the set of axioms, an ontology may be named using an IRI, and different versions of the same ontology may additionally be named with a version IRI. An ontology may also import other OWL 2 ontologies identified by their ontology or version IRIs. The set of axioms that constitute an ontology is taken to be equal to the union of the set of axioms contained in the ontology and the sets of axioms that constitute each of the ontologies that it imports; this is sometimes referred to explicitly as the *imports closure* of an ontology. Note that this definition is recursive: the imported ontologies could themselves import other ontologies, and so on.

Finally, an ontology can also be annotated with information such as the creator's name and copyright information. Annotation properties are used for this purpose, and there are several built-in annotation properties intended for use with ontologies; these include `owl:priorVersion`, `owl:backwardCompatibleWith`, and `owl:incompatibleWith`, all used to specify prior versions of the ontology, and optionally to describe their compatibility with the current ontology. More will be said about annotations and annotation properties in [▶ Sect. 9.3.6](#).

9.3.2 OWL 2 Datatypes

OWL 2 uses datatypes from XML Schema datatypes, so `xsd:integer` is a datatype in OWL, namely the type of integers; datatype restrictions using facets are also allowed, as in

`xsd:integer xsd:minInclusive "5"^^xsd:integer` (the integers greater than or equal to 5). The set of supported datatypes and facets is defined in the OWL 2 datatype map [59].

For data values (integers, strings, etc.) OWL 2 uses the syntax of RDF, so `"2"^^xsd:integer` is the way to write the integer 2, and `"23.5"^^xsd:decimal` is the way to write a decimal number. To enhance human readability for typical data values, the OWL 2 Manchester syntax allows strings, integers, decimals, and floats to be written as in most programming languages, as in `"abc"`, `25`, `25.55`, and `25.55F`. OWL 2 also allows plain RDF literals, which are a combination of a string and an optional language tag and can be written as in RDF, for example, `"favor"@en-us`. These plain literals belong to the datatype `rdf:PlainLiteral`.

9.3.3 OWL 2 Entities

As mentioned above, OWL 2 uses IRIs as names for classes, properties, individuals, and datatypes; collectively, these names are known as *entities*. In a Description Logic, the set of entities occurring in an ontology is usually called the signature. The entities, together with data values, make up the basic building blocks of OWL 2 ontologies.

In OWL 2, the properties are divided into three disjoint sets of object properties, data properties, and annotation properties. Object properties are used to relate one individual to another; for example, the object property `ex:worksFor` might be used to relate a person to a company. Data properties are used to relate an individual to a data value; for example, the data property `ex:hasAge` might be used to relate a person to an integer value. Finally, annotation properties are used to add uninterpreted information (such as textual comments) to individuals, classes, properties, and ontologies; more will be said about annotations in [Sect. 9.3.6](#).

Declarations and typing The entities used in an OWL 2 ontology can, and in some cases must, be typed using a suitable declaration. Declarations are used to avoid possible ambiguities and to ensure the required separation of object, data, and annotation property names. For example, declarations could be used as follows to state that `ex:worksFor`, `ex:hasAge` and `ex:authoredBy` are, respectively, object, data, and annotation properties:

```
ObjectProperty: ex:worksFor
DataProperty: ex:hasAge
AnnotationProperty: ex:authoredBy
```

Punning In OWL a given name could be used to refer to only a single type of entity. In contrast, OWL 2 allows the same name to be used to refer to different types of entity; for example, `ex:Eagle` might be used to denote both the class (a subclass of `ex:Bird`) and an individual (an instance of `ex:Species`):

```

Class: ex:Eagle
  SubClassOf: ex:Bird
Individual: ex:Eagle
  Types: ex:Species

```

On the face of it, this would seem to take OWL 2 beyond the strictly first-order realm of Description Logics. However, the semantics of OWL 2 is designed so that the interpretations of a name used as different entity types are totally unconnected; the interpretation of `ex:Eagle` the individual is, for example, totally unconnected to the interpretation of `ex:Eagle` the class. In effect, entity names are treated as though their type is part of the name, so the above example could be read as

```

Class: ex:Eaglec
  SubClassOf: ex:Bird
Individual: ex:Eaglej
  Types: ex:Species

```

This reuse of names, but with different meanings, is called *punning*. OWL 2 does place some restrictions on the use of punning: as discussed in [▶ Sect. 9.3.3](#), punning between object, data, and annotation properties is not allowed; in addition, punning between classes and datatypes is disallowed.

9.3.4 Expressions

As in a Description Logic, OWL 2 class and property expressions generalize classes and properties, while data ranges generalize datatypes. In particular, a property is a property expression, and property expressions can be combined using various operators to form new property expressions; a datatype is a data range, and data ranges can be combined using various operators (including facet-based restrictions) to form new data ranges; and a class is a class expression, and class expressions, property expressions, and data ranges can be combined using various operators to form new class expressions.

Property Expressions An object property expression in OWL 2 can be either an object property name or an expression constructed using the available operators as shown in [▶ Fig. 9.3](#), where P is an object property name (an IRI), and R is an arbitrary property expression. As can be seen, there is only one constructor for use with object property expressions: inverse. Arbitrary nesting of inverses is in principle possible,

Manchester Syntax	DL Syntax
P	P
$\text{inverse } R$	R^-

■ Fig. 9.3

OWL 2 object property expressions

Manchester Syntax	DL Syntax
B	B
D_1 and ... and D_n	$D_1 \cap \dots \cap D_n$
D_1 or ... or D_n	$D_1 \cup \dots \cup D_n$
not D	$\neg D$
$\{v_1 \dots v_n\}$	$\{v_1\} \cup \dots \cup \{v_n\}$

■ Fig. 9.4

OWL 2 data ranges

but as *inverse* (*inverse* R) is equivalent to R , it is reasonable to assume that all OWL 2 object property expressions are of the form P or *inverse* P for P an object property name (an IRI).

The set of constructors available for forming datatype properties is even more limited: there are none! All datatype property expressions are, therefore, of the form U for U a datatype property name (an IRI).

Data Ranges In OWL 2, the supported XML schema datatypes (including datatype restrictions using facets) are data ranges, and data ranges can also be formed from other data ranges and values using various constructors as shown in [Fig. 9.4](#), where B is an XML schema datatype or datatype restriction, D (possibly subscripted) is an arbitrary data range, and v_i is a data value. For example, a data range could be defined by using an XML schema facet to restrict a base type, as in `xsd:integer xsd:maxExclusive "10"^^xsd:integer` (the integers less than 10); it could be defined by enumerating a set of values, as in `{"1"^^xsd:integer "2"^^xsd:integer "3"^^xsd:integer}` (the integers 1, 2, and 3); or it could be defined by using one of the Boolean operators to combine other data ranges, as in `xsd:integer or xsd:float` (the union of the integers and floats).

Class Expressions As mentioned above, OWL 2 is very closely related to *SR_{OIQ}(D)*, and provides a correspondingly wide range of operators for building class expressions. These are summarized in [Fig. 9.5](#), where A is a class name (an IRI), C (possibly subscripted) is an arbitrary class expression, o_i is an individual name (an IRI), R is an object property expression, U is a datatype property expression, D is a data range, v is a data value, and n is a nonnegative integer. These can be used to form descriptions characterizing sets of individuals. For example, `ex:Person` and `ex:hasChild some ex:Person` describes those individuals that are instances of `ex:Person` and are related via the property `ex:hasChild` to an instance of `ex:Person` (i.e., parents).

Like *SR_{OIQ}(D)*, OWL 2 supports the standard Boolean constructors (`and`, `or`, and `not`), which correspond directly to \sqcap , \sqcup and \neg in Description Logic. The OWL 2 `OneOf` constructor allows a class to be formed by enumerating its instances, written $\{o_1 \dots o_n\}$ in the Manchester Syntax, and equivalent to a disjunction of nominals $\{o_1\} \sqcup \dots \sqcup \{o_n\}$ in *SR_{OIQ}(D)*. The built-in `owl:Thing` and `owl:Nothing` classes correspond directly to \top and \perp in Description Logic.

OWL 2 also supports the full set of *SR_{OIQ}(D)* restrictions, including `exists` and value restrictions (`some` and `only` in the Manchester Syntax) and both qualified and

Manchester Syntax	DL Syntax
<i>A</i> owl:Thing owl:Nothing	<i>A</i> ⊤ ⊥
<i>C</i> ₁ and ... and <i>C</i> _{<i>n</i>} <i>C</i> ₁ or ... or <i>C</i> _{<i>n</i>} not <i>C</i> { <i>o</i> ₁ ... <i>o</i> _{<i>n</i>} }	$C_1 \sqcap \dots \sqcap C_n$ $C_1 \sqcup \dots \sqcup C_n$ $\neg C$ $\{o_1\} \sqcup \dots \sqcup \{o_n\}$
<i>R</i> some <i>C</i> <i>R</i> only <i>C</i> <i>R</i> value <i>o</i> <i>R</i> self <i>R</i> min <i>n</i> <i>R</i> max <i>n</i> <i>R</i> exactly <i>n</i> <i>R</i> min <i>n C</i> <i>R</i> max <i>n C</i> <i>R</i> exactly <i>n C</i>	$\exists R.C$ $\forall R.C$ $\exists R.\{o\}$ $\exists R.self$ $\geq n R$ $\leq n R$ $\geq n R \sqcap \leq n R$ $\geq n R.C$ $\leq n R.C$ $\geq n R.C \sqcap \leq n R.C$
<i>U</i> some <i>D</i> <i>U</i> only <i>D</i> <i>U</i> value <i>v</i> <i>U</i> min <i>n</i> <i>U</i> max <i>n</i> <i>U</i> exactly <i>n</i> <i>U</i> min <i>n D</i> <i>U</i> max <i>n D</i> <i>U</i> exactly <i>n D</i>	$\exists U.D$ $\forall U.D$ $\exists U.\{v\}$ $\geq n U$ $\leq n U$ $\geq n U \sqcap \leq n U$ $\geq n U.D$ $\leq n U.D$ $\geq n U.D \sqcap \leq n U.D$

■ Fig. 9.5

OWL 2 class descriptions

unqualified maximum, minimum, and exact cardinality restrictions. Note that exact cardinality restrictions are equivalent to a symmetrical pair of Description Logic minimum and maximum cardinality restrictions, and that the Has Value restriction in OWL 2 (written *R* value *o* in Manchester Syntax) is simply shorthand for an exists restriction with a nominal as the restricting class. A similar set of restrictions can be used with datatypes and data values.

9.3.5 Axioms

OWL 2 axioms provide information about classes, properties, data ranges, keys, and individuals, as shown in [▶ Fig. 9.6](#), where *A* is a class name (an IRI), *C*

Manchester Syntax	DL Syntax
Class: A SubClassOf: C	$A \sqsubseteq C$
Class : A EquivalentTo: C	$A \equiv C$
EquivalentClasses: C_1, \dots, C_n	$C_i \equiv C_{i+1}$ for $1 \leq i < n$
DisjointClasses: C_1, \dots, C_n	$C_i \sqsubseteq \neg C_j$ for $1 \leq i < j \leq n$
Class: A DisjointUnionOf: C_1, \dots, C_n	$C_i \sqsubseteq \neg C_j$ for $1 \leq i < j \leq n$ $A \equiv C_1 \sqcup \dots \sqcup C_n$
ObjectProperty: P SubPropertyOf: R	$P \sqsubseteq R$
ObjectProperty: P EquivalentTo: R	$A \equiv C$
EquivalentProperties: R_1, \dots, R_n	$R_i \equiv R_{i+1}$ for $1 \leq i < n$
DisjointProperties: R_1, \dots, R_n	$R_i \sqsubseteq \neg R_j$ for $1 \leq i < j \leq n$
ObjectProperty: P InverseOf: R	$P \equiv R^{-}$
ObjectProperty: P Domain: C	$\exists P.T \sqsubseteq C$
ObjectProperty: P Range: C	$T \sqsubseteq \forall P.C$
ObjectProperty: P Characteristics: Functional	$T \sqsubseteq \leq 1P$
ObjectProperty: P Characteristics: InverseFunctional	$T \sqsubseteq \leq 1P^{-}$
ObjectProperty: P Characteristics: Reflexive	$T \sqsubseteq \exists P.self$
ObjectProperty: P Characteristics: Irreflexive	$\exists P.self \sqsubseteq \perp$
ObjectProperty: P Characteristics: Symmetric	$P \equiv P^{-}$
ObjectProperty: P Characteristics: Asymmetric	
ObjectProperty: P Characteristics: Transitive	$P \circ P \sqsubseteq P$
DataProperty: T SubPropertyOf: S	$T \sqsubseteq S$
DataProperty: T EquivalentTo: S	$A \equiv C$
EquivalentProperties: S_1, \dots, S_n	$S_i \equiv S_{i+1}$ for $1 \leq i < n$
DisjointProperties: S_1, \dots, S_n	$S_i \sqsubseteq \neg S_j$ for $1 \leq i < j \leq n$
DataProperty: T Domain: C	$\exists T.T_D \sqsubseteq C$
DataProperty: T Range: D	$T \sqsubseteq \forall T.D$
DataProperty: T Characteristics: Functional	$T \sqsubseteq \leq 1T$
Datatype: B EquivalentTo: D	$B \equiv D$
Class: A HasKey: U_1, \dots, U_n	U_1, \dots, U_n key for A
SameIndividual: i_1, \dots, i_n	$i_j = i_{j+1}$ for $1 \leq i < n$
DifferentIndividuals: i_1, \dots, i_n	$i_j \neq i_j$ for $1 \leq i < j \leq n$
Individual: i Types: C	$i : C$
Individual: i_1 Facts: $P i_2$	$\langle i_1, i_2 \rangle : P$
Individual: i_1 Facts: not $P i_2$	$i_1 : (\neg \exists P.\{i_2\})$
Individual: i Facts: $T v$	$\langle i, v \rangle : T$
Individual: i Facts: not $T v$	$i : (\neg \exists T.\{v\})$

■ Fig. 9.6

OWL 2 DL axioms and facts

(possibly subscripted) is an arbitrary class, P is an object property name (an IRI), R (possibly subscripted) is an arbitrary object property, T is a data property name (an IRI), S (possibly subscripted) is an arbitrary data property, D is a data range, B is a datatype (an IRI), U (possibly subscripted) is a property (either object or data), and i (possibly subscripted) is an individual. The axioms as presented here mirror the structural

specification and the RDF/XML and XML syntaxes; using the Manchester Syntax, however, it is also possible to group together statements about classes, properties, and individuals. For example, the following “class frame” could be used to define cricket fans as people who like cricket while at the same time asserting that cricket fans drink nothing but beer and that no individual can be both a cricket fan and a baseball fan:

```
Class: CricketFan
  EquivalentTo: ex:Person that ex:likes some ex:Cricket
  SubClassOf: ex:drinks only ex:Beer
  DisjointWith: ex:BaseballFan
```

This would be equivalent to the DL axioms:

```
CricketFan  $\equiv$  ex:Person  $\sqcap$   $\exists$ ex:likes.ex:Cricket
CricketFan  $\sqsubseteq$   $\forall$ ex:drinks.ex:Beer
CricketFan  $\sqsubseteq$   $\neg$ ex:BaseballFan
```

Similarly, statements about a given individual could be combined in Manchester Syntax as follows:

```
Individual: Canada
  Types: ex:Country
  Facts: ex:hasBorderWith USA,
         ex:hasPopulation 33487208,
         ex:hasLandArea 9093507
```

Asserting one class to be a subclass of or equivalent to another, **OWL 2** also allows for a set of classes to be asserted to be either equivalent or pairwise disjoint. Moreover, a class can be asserted to be equivalent to the disjoint union of a set of classes. As can be seen in [Fig. 9.6](#), all of these are “syntactic sugar” – they could be replaced with suitable sets of subclass or equivalence axioms.

Asserting one object property to be a subclass of or equivalent to another, and for equivalence and disjointness to be asserted for sets of object properties, OWL 2 also allows for a range of characteristics to be asserted for a given object property. These include asserting that the property is functional, inverse functional, reflexive, irreflexive, symmetric asymmetric, and/or transitive. Moreover, the inverse of a property can be given, as well as its range and domain.

For data properties, the range of available axioms is reduced: there is no inverse axiom, and the only characteristic that can be asserted is functionality. These restrictions are necessary in order to maintain a strict separation between reasoning about classes/individuals and reasoning about data ranges/values. Such a separation has been shown to allow for relatively simple integration of DL reasoners with datatype reasoners, where the datatype reasoner is used by the DL reasoner as an oracle able to answer relatively simple questions about data ranges and values [42,49].

OWL 2 also allows new datatypes to be introduced as abbreviations for data ranges, a convenient feature if identical data ranges are used in many places in an ontology.

For example, the following axiom introduces the datatype `over18` and defines it to be equivalent to the integers greater than 18:

```
Datatype: ex:over18 EquivalentTo: integer [> 18]
```

One of the new features of OWL 2 is keys, and these can be introduced using a suitable axiom. For example, the following axiom states that the combination of nationality and passport number is a key for persons:

```
Class: ex:Person HasKey: ex:hasNationality, ex:hasPassportNumber
```

where `ex:hasNationality` and `ex:hasPassportNumber` are data properties. This means that no two *named* individuals can have the same nationality and passport number.

Finally, OWL 2 allows for sets of individuals to be asserted to be the same (different names for the same object) or pairwise different (no two individuals name the same object), for individuals to be asserted to be instances of one or more classes, and for both positive and negative assertions about relationships between pairs of individuals. The first two statements are provided because OWL does not make any unique name assumption (UNA), that is, it is perfectly possible for `ex:USA` and `ex:UnitedStatesOfAmerica` to be two different names for the same object; it is therefore useful to be able to assert that a given set of names all refer to the same object, or to assert that UNA does apply to a given set of names, that is, that no two names from the set refer to the same object. For example, the following axioms could be used to assert the above mentioned equivalence between `ex:USA` and `ex:UnitedStatesOfAmerica`, and to assert that `ex:Alabama, ...`, `ex:Wyoming` all refer to different objects.

```
SameIndividual: USA, UnitedStatesOfAmerica  
DifferentIndividuals: ex:Alabama, ..., ex:Wyoming
```

9.3.6 Annotations

Annotations are a mechanism for adding extralogical “comments” to the ontology, that is, information that does not affect the formal meaning of the ontology and can thus be ignored by a reasoning system. Annotations could include, for example, human-readable labels, provenance, or hints on how the ontology should be displayed in a visualization tool. They are sometimes also used to capture information used in language extensions, for example, to associate a probability with an axiom in a probabilistic extension of OWL.

In OWL 2, it is possible to add annotations to almost any part of the ontology: they can be attached to the ontology itself, to entities such as classes, properties, and individuals, to class and property expressions, and to statements such as axioms and declarations – they can even be attached to other annotations. Annotations are specified by using an annotation property to associate the subject of the annotation with an annotation value, which

can be an IRI (which could, e.g., be a class or individual name), a string literal, or an anonymous individual. OWL 2 provides a number of predefined annotation properties: `rdfs:label`, `rdfs:comment`, `rdfs:seeAlso`, `rdfs:isDefinedBy`, `owl:deprecated`, `owl:versionInfo`, `owl:priorVersion`, `owl:backwardCompatibleWith`, and `owl:incompatibleWith`. In addition, some simple structuring of annotation types is provided for by allowing range, domain, and sub-property to be asserted for annotation properties.

This represents a significant advance over OWL, where only ontologies and entities could be annotated. In OWL, this restriction was imposed due to the difficulty of representing annotated statements in RDF, the problem being that RDF has no mechanism for using a triple or a set of triples as the subject of another triple. In OWL 2, this problem has been overcome by using a form of reification for annotated statements when rendered in the RDF syntax.

Some of the uses of annotation are illustrated in the following two examples. The first example illustrates an entity (the individual `ex:Canada`) annotated with a textual comment, and the comment itself being annotated with provenance information stating that the source is the CIA World Fact Book:

```
Individual: ex:Canada
Annotations:
  Annotation: ex:source ex:CIA-World-Fact-Book
             rdfs:comment
             "Situated in Northern North America, bordering
             the North Atlantic Ocean on the east, North
             Pacific Ocean on the west, and the Arctic Ocean
             on the north, north of the conterminous US"
```

The second example illustrates an axiom being annotated with provenance information (again the CIA World Fact Book) and with information about when the axiom was last updated:

```
Individual: ex:USA
Annotation: rdfs:label "United States of America"
Facts:
  Annotation: ex:source ex:CIA-World-Fact-Book,
             ex:lastUpdated "July 2009"
             ex:population 307212123,
```

9.3.7 Global Restrictions

In order to ensure that OWL 2 is a decidable language it is necessary to impose some global restrictions on the structure of ontologies. These restrictions correspond closely to

those used for the same purpose in the definition of $SR\mathcal{OIQ}(\mathbf{D})$ knowledge bases. The restrictions are called global because they depend on the ontology as a whole and not just on a single expression or axiom; for example, several of the restrictions relate to the property hierarchy, which depends on the set of property axioms occurring in the imports closure of the ontology. The definitions of some of the global restrictions are rather technical, and will only be sketched here; full details can be found in Sect. 11 of the OWL 2 Structural Specification and Functional-Style Syntax [59].

Firstly, it is necessary to distinguish *simple* properties. Roughly speaking, a property P is not simple if its existence is implied by a chain of other properties. This is the case if, for example, P is transitive, or if P has a sub-property S , and S is transitive. In the latter case, given individuals x , y , and z such that xSy and ySz , then from the transitivity of S it is possible to infer xSz , and from the fact that S is a sub-property of P it is possible to infer xPz . Intuitively, checking cardinality constraints for a non-simple property P is far more difficult because it is necessary to count not only directly related individuals but also those related via some chain of properties that implies P ; in fact this leads to undecidability in $SR\mathcal{OIQ}(\mathbf{D})$. Therefore, only simple roles can be used in cardinality restrictions. For similar reasons, only simple roles can be used in `self` restrictions, and in Functional, InverseFunctional, Irreflexive, Asymmetric, and Disjoint property axioms.

Secondly, the structure of property chains is restricted in various ways. They must, for example, satisfy an acyclicity condition, which is again needed in order to ensure decidability.

Thirdly, various restrictions are placed on the use of data ranges and datatype definition axioms. In particular, datatype definitions must be unique and acyclic; that is, given an axiom of the form:

Datatype: B EquivalentTo: D

where B is a newly defined datatype and D is a data range, D must not use B either directly or indirectly. This restriction means that datatype definitions can be treated as macros and simply “unfolded” by recursively substituting every occurrence of a defined datatype with the data range used to define it. In Description Logics, similar restrictions on concept definitions result in an *unfoldable* (sometimes called *definitorial*) TBox – one that can be eliminated by unfolding definitions into the ABox [1].

Fourthly, the use of anonymous individuals in axioms is restricted: they are not allowed to occur at all in Same Individual or Different Individuals axioms, or in negated individual facts, and their use in other axioms must satisfy another form of acyclicity constraint. These restrictions are designed to ensure that anonymous individuals can be eliminated from the ontology using a “rolling up” technique similar to the one used in conjunctive query answering [18].

Finally, the IRIs used to name ontologies and entities in OWL 2 must not be from the *reserved vocabulary*, that is, they must not be one of the IRIs used by the language itself. These include all of the IRIs with prefixes `rdf :`, `rdfs :`, `xsd :`, and `owl :`.

9.4 Semantics for OWL 2

In common with Description Logics, OWL 2 has a (first-order) model-theoretic semantics called the OWL 2 Direct Semantics [53]. This semantics is basically equivalent to simply translating the ontology into a $SR\mathcal{OIQ}(\mathcal{D})$ knowledge base as described in [Sect. 9.3](#) and then applying the standard Description Logic semantics.

This model-theoretic semantics is the ultimate arbiter of the meaning of OWL 2 constructs. However, it is generally sufficient to understand the informal meaning as described above, and as described in OWL 2 user facing documents such as the Primer [57]. For example, an individual is an instance of the intersection C and D exactly when it is an instance of both C and D , it is an instance of a restriction $P \text{ some } C$ exactly when it is related via the property P to an instance of C , and it is an instance of a restriction $U \text{ value } v$ exactly when it is related via the data property U to the value v . Similarly, an axiom `Class: A SubClassOf: C` implies that every instance of A is also an instance of C , while `Class: A EquivalentTo: C` additionally implies that every instance of C is an instance of D .

OWL 2 includes datatypes, and these are integrated into the language in much the same way as in Description Logics that include datatypes, in particular $SH\mathcal{OQ}(\mathcal{D})$. However, the particular datatypes used in OWL 2 are taken from RDF and XML Schema Datatypes [9], and inherit the semantics from the relevant specifications. Datatypes, such as `xsd:integer` and data values such as `"44"^^xsd:integer` are thus given the same meaning as they have in XML Schema. For example, the interpretation domains of `xsd:integer` and `xsd:float` are disjoint, so in OWL 2 `"1"^^xsd:integer` and `"1"^^xsd:float` are interpreted as two different values.

9.4.1 OWL 2 RDF-Based Semantics

For ontologies that use the RDF syntax, an alternative semantic account can be given by extending the RDF model theory with new conditions that capture the meaning of the OWL 2 vocabulary as described in the OWL 2 RDF-Based semantics [54]. For example, if an OWL 2 ontology in RDF syntax includes the triple

$$\langle C \text{ owl:complementOf } D \rangle$$

for C and D classes, then $\text{ICEXT}(C) = \text{IR} \setminus \text{ICEXT}(D)$, that is, the individuals that are instances of C ($\text{ICEXT}(C)$) must be equal to the whole of the interpretation domain (IR) minus the individuals that are instances of D ($\text{ICEXT}(D)$). Note the similarity to the semantics of negation in Description Logic presented in [Fig. 9.1](#); in fact, the RDF-based semantics is equipped with a correspondence theory that precisely defines the relationship between the two semantics [54].

In practice, the main difference between the Direct semantics and the RDF-based semantics is that the latter can be applied to RDF graphs that do not respect the various restrictions on OWL 2 syntax described in [Sect. 9.3](#); indeed, the RDF-based semantics

can be applied to arbitrary RDF graphs. It is important to be aware, however, that additional meaning (beyond that derived from the RDF semantics [24]) is only given to those parts of the graph that correspond to OWL 2 constructions as described in [Sect. 9.3](#). It therefore makes more sense to think of the RDF-based semantics as a semantics for OWL 2 ontologies that do not respect the restrictions required by OWL 2 DL.

This style of usage of OWL 2 is known as OWL 2 Full. Although this easing of language restrictions in OWL 2 Full can sometimes be convenient, it also has the effect of making the language undecidable [33,47]; this makes it difficult to provide reasoning systems, and impossible to provide systems that can guarantee to correctly answer arbitrary queries. In the case of OWL Full, this resulted in applications typically using an ad hoc subset of the language along with some simple inference rules to provide reasoning that was sound but incomplete, an approach that was undesirable both from the point of view of reliability and interoperability. In OWL 2, the OWL 2 RL profile is designed in part to overcome this problem by defining a suitable subset, providing an axiomatization that can be used as the basis for a rule-set, and describing conditions on the ontology under which an implementation based on such a rule set would be sound and complete. It is conjectured that OWL 2 Full will in effect be OWL 2 RL under the RDF-based semantics.

It is also worth pointing out that the new features of OWL 2 make it much easier for applications to live with the restrictions imposed in OWL 2 DL. In particular, punning in OWL 2 means that class/individual meta-modeling is now possible in OWL 2 DL, and keys can be used instead of applying inverse functionality to data properties – another common reason for ontologies to be OWL Full.

9.5 OWL 2 Profiles

As noted above, OWL 2 DL is closely related to $\mathcal{SROIQ}(\mathbf{D})$, a very expressive Description Logic. $\mathcal{SHOIN}(\mathbf{D})$ is (potentially) very difficult to reason with, as standard inference problems, such as Ontology Consistency, Class Expression Satisfiability, Class Expression Subsumption, and Instance Checking, all have 2NExpTime complexity.

For these reasons, three profiles have been defined: language fragments that have desirable computational properties, and in particular lower worst-case complexities for the above-mentioned inference problems (see <http://www.w3.org/TR/owl2-profiles/>). These profiles are called OWL 2 EL, OWL 2 QL, and OWL 2 RL. Note that the old OWL Lite profile has been deprecated as it provides no significant computational advantage (OWL Lite ontologies would, in general, be considered standard OWL 2 ontologies).

9.5.1 OWL 2 EL

OWL 2 EL is based on the \mathcal{EL}^{++} , a Description Logic for which standard reasoning problems can be performed in time, that is, polynomial with respect to the size of the

ontology. This profile captures the expressive power used by many ontologies used in the life sciences, and is particularly useful in applications employing ontologies that contain very large numbers of properties and/or classes, as is often the case with life science ontologies.

Unlike the other two profiles, EL has the advantage of being “symmetrical,” in the sense that the restrictions apply equally to class expressions occurring on the left-hand side and the right-hand side of class inclusion axioms, making it very easy to define and to use. The restrictions on class expressions rule out the use of universal quantification (i.e., `ObjectAllValuesFrom` and `DataAllValuesFrom`), cardinality restrictions, disjunction, negation, enumerations involving multiple individuals (i.e., `ObjectOneOf` and `DataOneOf`), and most property characteristics (including `InverseObjectProperties`, `DisjointObjectProperties`, `DisjointDataProperties`, `IrreflexiveObjectProperty`, `FunctionalObjectProperty`, `InverseFunctionalObjectProperty`, `SymmetricObjectProperty`, and `AsymmetricObjectProperty`). In addition, the set of supported datatypes has been designed such that the intersection of the value spaces of any set of datatypes is either empty or infinite, which is necessary to retain the desired computational properties. As a result, use of the following datatypes is ruled out: `xsd:double`, `xsd:float`, `xsd:nonPositiveInteger`, `xsd:positiveInteger`, `xsd:negativeInteger`, `xsd:long`, `xsd:int`, `xsd:short`, `xsd:byte`, `xsd:unsignedLong`, `xsd:unsignedInt`, `xsd:unsignedShort`, `xsd:unsignedByte`, `xsd:language`, and `xsd:boolean`. Finally, EL also rules out the use of anonymous individuals.

Several reasoners are available for OWL 2 EL, including CB [36], CEL [4], Pellet (in fact Pellet supports all of OWL 2, but includes a dedicated EL reasoner for optimized reasoning with this profile) [74], and Snorocket [14,39]. These reasoners all use a saturation-based technique in which the TBox is extended to explicitly include all subsumption relationships holding between named classes. These algorithms are extremely effective at dealing with large ontologies: the CB reasoner can, for example, fully classify the 400,000 class SNOMED-CT ontology [76] in less than 60 s. Recent work has shown how scalability can also be extended to large datasets by using database technology to store the set of individual axioms (the ABox) and employing a mixture of materialization and query rewriting [44].

9.5.2 OWL 2 QL

OWL 2 QL is based on $DL\text{-Lite}_R$, a Description Logic for which conjunctive query answering can be implemented using conventional relational database systems and so can be performed in LOGSPACE with respect to the size of the data (individual axioms). It is aimed at applications that use very large volumes of instance data, and where query answering is the most important reasoning task. As in OWL 2 EL, polynomial time algorithms can be used to implement the ontology consistency and class expression subsumption reasoning problems. The expressive power of OWL 2 QL is necessarily quite limited, although it does include most of the main features of conceptual models such as UML class diagrams and ER diagrams.

Several variants of DL-Lite have been described in the literature, with OWL 2 QL being based on the DL-Lite_R variant. This has the advantage that, although the instance data is assumed to be in a relational database, no unique name assumption (UNA) is required – this is because the UNA would have no impact on the semantic consequences of a DL-Lite_R ontology. OWL 2 QL not only restricts the kinds of class expression that can be used, but also varies these restrictions depending where the expression occurs in an axiom (e.g., as the subclass or superclass part of a `subClassOf` axiom. This makes the precise definition of the profile rather complex, and the reader is referred to the OWL 2 Profiles specification for full details [58]. The set of supported datatypes is the same as for OWL 2 EL, and the use of anonymous individuals is similarly ruled out. Finally, OWL 2 QL does not support individual equality assertions (`SameIndividual`), because this would make queries no longer first-order rewritable, with the result that query answering could no longer be implemented directly using relational database technologies. However, individual equality could be materializing the equality relation in the database and using the resulting relation in query answering [69].

Several reasoners are available for DL-Lite_R/OWL 2 QL, including OwlGres (see <http://pellet.owldl.com/owlgres>), and QuOnto (see <http://www.dis.uniroma1.it/~quonto/>). Both of these reasoners are based on query rewriting techniques that transform a conjunctive query against the ontology into a set of queries against the individual axioms (the ABox) only, and ultimately (via mappings from ontology class and property names to SQL queries) into SQL queries against a database (or databases) where the instance data are stored [65].

9.5.3 OWL 2 RL

The OWL 2 RL profile is aimed at applications that require scalable reasoning without sacrificing too much expressive power. It is designed to accommodate both OWL 2 applications that can trade the full expressivity of the language for efficiency, and RDF(S) applications that need some added expressivity from OWL 2. This is achieved by defining a syntactic subset of OWL 2, which is amenable to implementation using rule-based technologies, and presenting a partial axiomatization of the OWL 2 RDF-based semantics in the form of first-order implications that can be used as the basis for such an implementation (see [▶ Sect. 9.4.1](#)). The design of OWL 2 RL was inspired by Description Logic Programs [20] and pD* [79].

Like OWL 2 QL, the syntax of RL is asymmetric in the sense that different constraints apply to class expressions depending where they occur in ontology axioms. Essentially, this means allowing enumerations (`ObjectOneOf`), intersections (`ObjectIntersectionOf`), unions (`ObjectUnionOf`), and existential restrictions in the subclass position of a `subClassOf` axiom, and intersection (`ObjectIntersectionOf`), negation (`ObjectComplementOf`), universal restrictions (`ObjectAllValuesFrom` and `DataAllValuesFrom`), existential restrictions using an individual or data value (`ObjectHasValue` and `DataHasValue`), and at most 0/1 cardinality restrictions (`ObjectMaxCardinality` and `DataMaxCardinality`

with values 0 or 1) in the superclass position of a `subClassOf` axiom (see the OWL 2 Profiles specification for full details [58]).

For ontologies satisfying the above-mentioned syntactic constraints, a suitable rule-based implementation (e.g., one based on the partial axiomatization of the RDF-based semantics) will have desirable computational properties; for example, it can return all and only the correct answers to ground atomic queries (see Theorem PR1 from the OWL 2 Profiles specification [58] and the OWL 2 Conformance specification [52]). As mentioned in Sect. 9.4.1, such an implementation can also be used with arbitrary RDF graphs. In this case, however, the above-mentioned computational properties no longer hold – in particular, it is no longer possible to guarantee that all correct answers can be returned, for example, if the RDF graph uses the built-in vocabulary in unusual ways.

Several reasoners are available for OWL 2 RL, including Elly (see <http://elly.sourceforge.net/>), Jena (see <http://jena.sourceforge.net/>), and the Oracle Database 11g OWL Reasoner (see http://www.oracle.com/technology/tech/semantic_technologies/index.html). These implementations are all based on rule-extended triple stores and relational databases and work by computing all “relevant” inferences and materializing them in the store/database. This may require significant additional time and storage, but if this is within acceptable bounds, then queries can subsequently be answered simply by querying the store/database. Completeness is, however, dependent on the set of materialized inferences and the kind of query being answered and, as mentioned above, can only be guaranteed for ground atomic queries against ontologies satisfying the syntactic restrictions on the RL profile.

9.6 OWL Tools and Applications

As discussed in Sect. 9.2, one of the motivations for basing OWL on a description logic was the ready availability of tools and infrastructure. Similarly, in the case of OWL 2, the willingness and ability of tool developers to support the language was an important influence on its design.

Regardless of the target ontology language, developing good quality ontologies is an extremely difficult and time-consuming task. It is therefore essential to provide ontology engineers with tool support. A range of ontology development tools are available for this purpose, including Swoop [34], PROTÉGÉ [38], and TopBraid Composer (see <http://www.topbraidcomposer.com/>).

Ontology development tools invariably use a DL reasoner to provide feedback to the user about the logical implications of their design, for example, to warn them about inconsistencies. Moreover, reasoners are an essential feature of applications where they are required, for example, in order to answer both conceptual and data retrieval queries.

The availability of tools and reasoning systems has been an important factor in the increasingly widespread use of OWL. Applications of OWL are particularly prevalent in the life sciences where it has been used by the developers of several large biomedical ontologies, including the SNOMED, GO, and BioPAX ontologies mentioned above, the

Microarray Gene Expression Data (MGED) ontology (see <http://mged.sourceforge.net/ontologies/index.php>), the Foundational Model of Anatomy (FMA) [19], and the National Cancer Institute thesaurus (NCI) [23].

Another important component in OWL applications, including the above mentioned editors, is the de facto standard Manchester OWL API [25]. The API takes care of parsing and rendering OWL ontologies in various different syntaxes, and also provides a standard interface to OWL reasoners. This means that OWL applications can easily switch between reasoners, choosing whichever proves to be most effective.

9.6.1 Ontology Design Tools

Ontologies are often large and complex: the well-known SNOMED clinical terms ontology includes, for example, more than 400,000 class names [75]. Building and maintaining such ontologies is very costly and time consuming, and providing tools and services to support this “ontology engineering” process is of crucial importance to both the cost and the quality of the resulting ontology. State-of-the-art ontology development tools, such as Swoop [34], PROTÉGÉ [38], and TopBraid Composer (see <http://www.topbraidcomposer.com/>), use a DL reasoner, such as FaCT++ [80], Hermit [51], Racer [22], or Pellet [74], to provide feedback to the user about the logical implications of their design.

The importance of reasoning support in ontology applications was highlighted in a recent paper describing a project in which the Medical Entities Dictionary (MED), a large ontology (100,210 classes and 261 properties) that is used at the Columbia Presbyterian Medical Center, was converted into OWL and checked using an OWL reasoner [37]. This check revealed “systematic modeling errors,” and a significant number of missed subClassOf relationships which, if not corrected, “could have cost the hospital many missing results in various decision support and infection control systems that routinely use MED to screen patients.”

Similarly, an extended version of the SNOMED ontology was checked using an OWL reasoner, and a number of missing subClassOf relationships were found. This ontology is being used by the UK National Health Service (NHS) to provide “A single and comprehensive system of terms, centrally maintained and updated for use in all NHS organizations and in research,” and as a key component of their multibillion pound “Connecting for Health” IT program. An important feature of this system is that it can be extended to provide more detailed coverage if needed by specialized applications. For example, a specialist allergy clinic may need to distinguish allergies caused by different kinds of nut, and so may add new terms to the ontology such as AlmondAllergy:

$$\text{AlmondAllergy} \equiv \text{Allergy} \sqcup \exists \text{causedBy} . \text{Almond}$$

Using a reasoner to insert this new term into the ontology will ensure that it is recognized as a subClassOf NutAllergy. This is clearly of crucial importance in order to ensure that patients with an AlmondAllergy are correctly identified in the national records system as patients having a NutAllergy.

As well as computing the class hierarchy, ontology design tools typically provide (at least) warnings about inconsistencies and redundancies. An inconsistent (sometimes called unsatisfiable) class is one whose description is “over-constrained,” with the result that it can never have any instances. This is typically an unintended feature of the design – why introduce a name for a class that can never have any instances – and may be due to subtle interactions between axioms. It is, therefore, very useful to be able to detect such classes and bring them to the attention of the ontology engineer. For example, during the development of an OWL ontology at the NASA Jet Propulsion Laboratory (see [▶ Sect. 9.6.2](#)), the class “OceanCrustLayer” was found to be inconsistent. This was discovered (with the help of debugging tools) to be the result of its being defined to be both a region and a layer, one of which (layer) was a two-dimensional (2D) object and the other a three-dimensional (3D) object, where the axioms describing 2D and 3D objects ensured that these two classes were disjoint (had no instances in common). The inconsistency thus highlighted a fundamental error in the design of the ontology, discovering, and repairing, which obviously improved the quality of the ontology.

It is also possible that the descriptions in the ontology mean that two classes necessarily have exactly the same set of instances, that is, that they are alternative names for the same class. This may be desirable in some situations, for example, to capture the fact that “Myocardial infarction” and “Heart attack” mean the same thing. It could, however, also be the inadvertent result of interactions between descriptions, and so it is also useful to be able to alert users to the presence of such “synonyms.” For example, when developing a medical terminology ontology a domain expert added the following two axioms:

$$\begin{aligned} \text{AspirinTablet} &\equiv \exists \text{ hasForm. Tablet} \\ \text{AspirinTablet} &\sqsubseteq \text{AspirinDrug} \end{aligned}$$

intending to capture the information that aspirin tablets are just those aspirin drugs that have the form of a tablet. Instead, these axioms had the effect of making *every* kind of tablet be an aspirin tablet. This was immediately corrected when the reasoner alerted the domain expert to the unexpected equivalence between `Tablet` and `AspirinTablet`.

As discussed above, ontology development tools usually check for implicit subsumption relationships, and update the class hierarchy accordingly. This is also a very useful design aid: it allows ontology developers to focus on class descriptions, leaving the computation of the class hierarchy to the reasoner, and it can also be used by developers to check if the hierarchy induced by the class descriptions is consistent with their intuition. This may not be the case when, for example, errors in the ontology result in unexpected subsumption inferences, or “under-constrained” class descriptions result in expected inferences not being found. The latter case is extremely common, as it is easy to inadvertently omit axioms that express “obvious” information. For example, an ontology engineer may expect the class of patients who have a fracture of both the tibia and the fibula to be a subclass of “patient with multiple fractures”; however, this may not be the case if the ontology does not include (explicitly or implicitly) the information that the tibia and fibula are different bones. Failure to find the expected subsumption relationship will alert the engineer to the missing `DisjointClasses` axiom.

Recent work has also shown how reasoning can be used to support modular design [16] and module extraction [15], important techniques for working with large ontologies. When developing a large ontology such as SNOMED, it is useful if not essential to divide the ontology into modules, for example, to facilitate parallel work by a team of ontology developers. Reasoning techniques can be used to alert the developers to unanticipated and/or undesirable interactions between the various modules. Similarly, it may be desirable to extract from a large ontology a smaller module containing all the information relevant to some subset of the domain, for example, heart disease – the resulting small(er) ontology will be easier for humans to understand and easier for applications to use. Reasoning can be used to compute a module that is as small as possible while still containing all the necessary information.

Finally, in order to maximize the benefit of reasoning services, tools should be able to explain inferences: without this facility, users may find it difficult to repair errors in the ontology and may even start to doubt the correctness of inferences. Explanation typically involves computing a (hopefully small) subset of the ontology that still entails the inference in question, and if necessary presenting the user with a chain of reasoning steps [35]. [Figure 9.7](#), for example, shows an explanation, produced by the PROTÉGÉ ontology development tool, of the above-mentioned inference with respect to the inconsistency of OceanCrustLayer.

9.6.2 Reasoning in Deployed Applications

Reasoning is also important when ontologies are deployed in applications – it is needed, for example, in order to answer structural queries about the domain and to retrieve data. For example, biologists use ontologies such as the Gene Ontology (GO) and the Biological Pathways Exchange ontology (BioPAX) to annotate data from gene sequencing experiments to be able to answer complex queries such as “what DNA binding products interact with insulin receptors.” Answering this query requires a reasoner not only to identify individuals that are (perhaps only implicitly) instances of DNA binding products and of

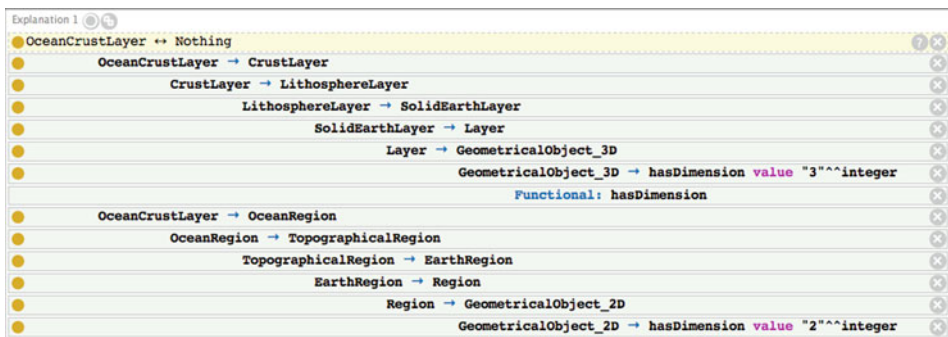


Fig. 9.7

An explanation from PROTEGE

insulin receptors, but also to identify which pairs of individuals are (perhaps only implicitly) related via the `interactsWith` property.

It is easy to imagine that, with large ontologies, query answering may be a very complex task. The use of DL reasoners allows OWL ontology applications to answer complex queries, and to provide guarantees about the correctness of the result. This is obviously of crucial importance when ontologies are used in safety critical applications such as medicine; it is, however, also important if ontology-based systems are to be used as components in larger applications, such as the Semantic Web, where the correct functioning of automated processes may depend on their being able to (correctly) answer such queries.

Ontologies are also widely used to facilitate the sharing and integration of information. The Neurocommons project (see <http://sciencecommons.org/projects/data/>), for example, aims to provide a platform for sharing and integrating knowledge in the neuroscience domain. A key component is an ontology of annotations that will be used to integrate available knowledge on the Web, including major neuroscience databases. Similarly, the OBO Foundry (see <http://www.obofoundry.org/>) is a library of ontologies designed to facilitate information sharing and integration in the biomedical domain, while the NCI ontology mentioned above constitutes the working vocabulary used in NCI data systems (see <http://ncicb.nci.nih.gov/NCICB/core/EVS/>).

In information integration applications, the ontology can play several roles: it can provide a formally defined and extensible vocabulary for use in semantic annotations, it can be used to describe the structure of existing sources and the information that they store, and it can provide a detailed model of the domain against which queries can be formulated. Such queries can be answered by using semantic annotations and structural knowledge to retrieve and combine information from multiple sources [78]. It should be noted that the use of ontologies in information integration is far from new, and has already been the subject of extensive research within the database community [5].

Other examples of OWL ontology applications include:

- United Nations Food and Agriculture Organization (FAO) is using OWL to develop a range of ontologies covering areas such as agriculture and fisheries (see <http://www.fao.org/agris/aos/Applications/intro.htm>).
- The Semantic Web for Earth and Environmental Terminology (SWEET) ontologies developed at the US National Aeronautics and Space Administration (NASA) Jet Propulsion Laboratory (see <http://sweet.jpl.nasa.gov/ontology/>). These include ontologies describing space, the biosphere, and the sun. SWEET is now being expanded by a number of earth and space science efforts, and has been augmented in the GEON project (see <http://www.geongrid.org/>) to cover the solid earth, and by the Virtual Solar Terrestrial Observatory Project (see <http://vsto.hao.ucar.edu/>) to include much more information on the atmosphere.
- An ontology used at General Motors in a project to help quality improvement activities for assembly line processes in different production sites [46].

9.7 Summary

OWL 2 exhibits the desirable features of Description Logics, including useful expressive power, formal syntax and semantics, decidability, and practical reasoning systems, resulting in OWL 2 providing effective ontology representation facilities. As well, OWL 2 is firmly a part of the W3C Semantic Web, with its use of IRIs for names, XML Schema datatypes, and ontologies as Web documents, which can then import other OWL 2 ontologies over the Web. OWL 2 thus firmly integrates ontologies into the Semantic Web.

It is not necessary to understand all of the formal aspects of OWL 2 in order to use OWL 2 effectively. All that is required is a reasonable understanding of what it means to define aspects of a class, property, or individual in an ontology. The use of OWL 2 ontology editors, such as *PROTÉGÉ*, serves as a bridge between the syntax of an ontology and its semantics, calling OWL 2 reasoners, such as *Pellet*, *Hermit*, and *Fact++* to determine consequences of the ontology statements and then present them in an easy-to-see fashion.

It is also not necessary to completely understand the differences between the various profiles of OWL 2. It is possible to design an ontology without reference to the profiles, and without checking which profile the ontology belongs to. Of course, if the computational or implementation benefits of a particular profile are needed, it is best to keep the ontology being designed within that particular profile. It is expected that ontology editors will soon be able to both check which profile an ontology is in, and impose syntactic constraints to ensure that an ontology stays within a given profile.

The situation is somewhat more complex with OWL 2 Full. First, determining whether an ontology is outside of OWL 2 DL has to be done on the entire ontology. Second, most OWL 2 tools do not handle OWL 2 Full, so using OWL 2 Full results in a loss of tool support. Third, reasoning in OWL 2 Full is undecidable, so there is no chance of effective reasoning tools being developed. The use of OWL 2 Full should thus be reserved for situations where there already is some existing RDFS that does not fit within OWL 2 DL and cannot be modified.

In practice, relatively few OWL Full applications have emerged to date, and where OWL Full ontologies are found, they often turn out to be outside the OWL DL subset only as the result of minor syntactic errors and thus should have been in OWL DL. Fragments of OWL 2 are, however, sometimes used as ad hoc extensions to RDFS. A common example is the use of OWL functional properties, and explicit equivalences and (in) equalities, in what would otherwise be an RDFS ontology. In many cases, the RDFS ontology can or should be in OWL 2 DL, but sometimes there is some significant aspect of the RDFS ontology that requires the use of OWL Full. The need to use OWL Full should be further reduced in OWL 2 due to support for punning and key constraints.

There remain, of course, significant issues that are not handled by OWL 2, but that are definitely pertinent to the Semantic Web.

- OWL 2 excludes useful expressive power to remain decidable, or just because there was not enough time in the OWL Working Group to specify the construct. In particular,

DL-safe rules [50] and Description Graphs [48] are not in OWL 2, even though there are Description Logics that include these features and even Description Logic reasoners that implement them (such as HermiT). Similarly, there are proposals for N-ary relations [13] and N-ary datatypes [60] in Description Logics but neither of these features are in OWL 2.

- OWL 2 is monotonic, as are most formal representation languages that can handle reasonably large amounts of information, and thus cannot handle default reasoning or localized closed-world assumptions.
- OWL 2 is binary, as are most formal representation languages that can handle reasonably large amounts of information, and thus cannot handle probabilistic or fuzzy reasoning.
- OWL 2 treats ontologies as single entities and does not allow the extraction of part of an ontology.

Some of the non-present features are already in particular proposals for extensions for OWL 2. If considered desirable by a sufficiently large community, it is likely that OWL 2 reasoners and other systems will implement them in a compatible fashion, leading to de facto extensions to OWL 2. Other non-present features, however, would require significant research to provide formal foundations, reasoning algorithms, and/or effective reasoners for them, which must be carried out before they can be included in future OWL 2 extensions.

9.8 Cross-References

- [KR and Reasoning on the Semantic Web: RIF](#)
- [KR and Reasoning on the Semantic Web: Web-scale Reasoning](#)
- [Ontologies and the Semantic Web](#)
- [Querying the Semantic Web: SPARQL](#)

References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F. (eds.): *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, Cambridge (2003)
2. Baader, F., Hanschke, P.: A schema for integrating concrete domains into concept languages. In: *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI 1991)*, Sidney, pp. 452–457 (1991)
3. Baader, F., Hollunder, B.: *KRIS: Knowledge Representation and Inference System*. SIGART Bull. 2(3), 8–14 (1991)
4. Baader, F., Lutz, C., Suntisrivaraporn, B.: CEL – a polynomial-time reasoner for life science ontologies. In: Furbach, U., Shankar, N. (eds.) *Proceedings of the Third International Joint Conferences on Automated Reasoning (IJCAR 2006)*, Seattle. *Lecture Notes in Artificial Intelligence*, vol. 4130, pp. 287–291. Springer, Berlin (2006).
5. Batini, C., Lenzerini, M., Navathe, S.B.: A comparative analysis of methodologies for database schema integration. *ACM Comput. Surv.* 18(4), 323–364 (1986)
6. Bechhofer, S., Goble, C., Horrocks, I.: DAML + OIL is not enough. In: *Proceedings of the First*

- International Semantic Web Working Symposium (SWWS 2001), Stanford, pp. 151–159. <http://www.semanticweb.org/SWWS/program/full/SWWSProceedings.pdf> (2001)
7. Bechhofer, S., Horrocks, I., Goble, C., Stevens, R.: OilEd: a reason-able ontology editor for the semantic web. In: Proceedings of the Joint German/Austrian Conferences on Artificial Intelligence (KI 2001), Vienna. Lecture Notes in Artificial Intelligence, vol. 2174, pp. 396–408. Springer, Berlin (2001)
 8. Beckett, D.: RDF/XML syntax specification (revised), W3C Recommendation. <http://www.w3.org/TR/rdf-syntax-grammar/> (2004)
 9. Biron, P.V., Malhotra, A.: XML schema part 2: datatypes, W3C Recommendation. <http://www.w3.org/TR/xmlschema-2/> (2001)
 10. Brachman, R.J.: A structural paradigm for representing knowledge. Ph.D. thesis, Harvard University, Cambridge (1977). Revised version published as BBN Report No. 3605. Bolt Beranek and Newman, Cambridge (1978)
 11. Brachman, R.J., Schmolze, J.G.: An overview of the KL-ONE knowledge representation system. *Cogn. Sci.* **9**(2), 171–216 (1985)
 12. Bresciani, P., Franconi, E., Tessaris, S.: Implementing and testing expressive description logics: preliminary report. In: Proceedings of the 1995 Description Logic Workshop (DL 1995), Rome, pp. 131–139 (1995)
 13. Calvanese, D., De Giacomo, G., Lenzerini, M.: Reasoning in expressive description logics with fixpoints based on automata on infinite trees. In: Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI 1999), Stockholm, pp. 84–89 (1999)
 14. Cornet, R., Spackman, K.A. (eds.): Proceedings of the Third International Conference on Knowledge Representation in Medicine (KR-MED 2008), Phoenix. CEUR, vol. 410. CEUR-WS.org (2008)
 15. Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: Just the right amount: extracting modules from ontologies. In: Proceedings of the 16th International World Wide Web Conference (WWW 2007), Banff (2007)
 16. Cuenca Grau, B., Kazakov, Y., Horrocks, I., Sattler, U.: A logical framework for modular integration of ontologies. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), Hyderabad, pp. 298–303 (2007)
 17. Fensel, D., van Harmelen, F., Horrocks, I., McGuinness, D., Patel-Schneider, P.F.: OIL: an ontology infrastructure for the semantic web. *IEEE Intell. Syst.* **16**(2), 38–45 (2001)
 18. Glimm, B., Horrocks, I., Lutz, C., Sattler, U.: Conjunctive query answering for the description logic *SHIQ*. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), Hyderabad, pp. 399–404 (2007)
 19. Golbreich, C., Zhang, S., Bodenreider, O.: The foundational model of anatomy in OWL: experience and perspectives. *J. Web Semant.* **4**(3), 181–195 (2006)
 20. Groszof, B.N., Horrocks, I., Volz, R., Decker, S.: Description logic programs: combining logic programs with description logic. In: Proceedings of the 12th International World Wide Web Conference (WWW 2003), Budapest, pp. 48–57. ACM, New York (2003)
 21. Grosso, W.E., Eriksson, H., Ferguson, R.W., Gennari, J.H., Tu, S.W., Musen, M.A.: Knowledge modelling at the millennium (the design and evolution of Protege-2000). In: Proceedings of the 12th Workshop on Knowledge Acquisition, Modeling and Management (KAW 1999), Banff (1999)
 22. Haarslev, V., MoUer, R.: RACER system description. In: Proceedings of the First International Joint Conference on Automated Reasoning (IJCAR 2001), Siena. Lecture Notes in Artificial Intelligence, vol. 2083, pp. 701–705. Springer, Berlin (2001)
 23. Hartel, F.W., de Coronado, S., Dionne, R., Frago, G., Golbeck, J.: Modeling a description logic vocabulary for cancer research. *J. Biomed. Inform.* **38**(2), 114–129 (2005)
 24. Hayes, P.: RDF model theory, W3C Recommendation. <http://www.w3.org/TR/rdf-mt/> (2004)
 25. Horridge, M., Bechhofer, S., Noppens, O.: Igniting the OWL 1.1 touch paper: the OWL API. In: Proceedings of the Third OWL Experiences and Directions Workshop (OWLED 2007), Innsbruck. CEUR, vol. 258. <http://ceur-ws.org/> (2007)
 26. Horridge, M., Drummond, N., Goodwin, J., Rector, A., Stevens, R., Wang, H.: The Manchester

- OWL syntax. In: Proceedings of the Second OWL Experiences and Directions Workshop (OWLED 2006), Athens. CEUR, vol. 216. <http://ceur-ws.org/> (2006)
27. Horrocks, I.: Using an expressive description logic: FaCT or fiction? In: Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR 1998), Trento, pp. 636–647 (1998)
 28. Horrocks, I., Fensel, D., Broekstra, J., Decker, S., Erdmann, M., Goble, C., van Harmelen, F., Klein, M., Staab, S., Studer, R., Motta, E.: OIL: the ontology inference layer. Technical report IR-479. Faculty of Sciences, Vrije Universiteit Amsterdam. <http://www.ontoknowledge.org/oil/> (2000)
 29. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible *SR_OT_Q*. In: Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR 2006), Lake District, pp. 57–67. AAAI Press, Menlo Park (2006)
 30. Horrocks, I., Patel-Schneider, P.F., van Harmelen, F.: From *SHIQ* and RDF to OWL: the making of a web ontology language. *J. Web Semant.* 1(1), 7–26 (2003)
 31. Horrocks, I., Sattler, U.: Ontology reasoning in the *SHOQ(D)* description logic. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001), Seattle, pp. 199–204. Morgan Kaufmann, Los Altos (2001)
 32. Horrocks, I., Sattler, U.: A tableaux decision procedure for *SHO_TQ*. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005), Edinburgh, pp. 448–453 (2005)
 33. Horrocks, I., Sattler, U., Tobies, S.: Practical reasoning for expressive description logics. In: Ganzinger, H., McAllester, D., Voronkov, A. (eds.) Proceedings of the Sixth International Conference on Logic for Programming and Automated Reasoning (LPAR 1999), Tbilisi. Lecture Notes in Artificial Intelligence, vol. 1705, pp. 161–180. Springer (1999)
 34. Kalyanpur, A., Parsia, B., Sirin, E., Cuenca-Grau, B., Hendler, J.: SWOOP: a web ontology editing browser. *J. Web Semant.* 4(2), 144–153 (2006)
 35. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.: Debugging unsatisfiable classes in OWL ontologies. *J. Web Semant.* 3(4), 243–366 (2005)
 36. Kazakov, Y.: Consequence-driven reasoning for horn SHIQ ontologies. In: Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009), Pasadena, pp. 2040–2045 (2009)
 37. Kershenbaum, A., Fokoue, A., Patel, C., Welty, C., Schonberg, E., Cimino, J., Ma, L., Srinivas, K., Schloss, R., Murdock, J. W.: A view of OWL from the field: use cases and experiences. In: Proceedings of the Second OWL Experiences and Directions Workshop, Athens. CEUR, vol. 216. <http://ceur-ws.org/> (2006)
 38. Knublauch, H., Fergerson, R., Noy, N., Musen, M.: The protege OWL plugin: an open development environment for semantic web applications. In: McIlraith, S. A., Plexousakis, D., van Harmelen, F. (eds.) Proceedings of the Third International Semantic Web Conference (ISWC 2004), Hiroshima. Lecture Notes in Computer Science, vol. 3298, pp. 229–243. Springer (2004)
 39. Lawley, M.: Exploiting fast classification of SNOMED CT for query and integration of health data. In: Cornet, R., Spackman, K. A. (eds.) Proceedings of the Third International Conference on Knowledge Representation in Medicine (KR-MED 2008), Phoenix. CEUR, vol. 410. CEUR-WS.org (2008)
 40. Levesque, H.J., Brachman, R.J.: Expressiveness and tractability in knowledge representation and reasoning. *Comput. Intell.* 3, 78–93 (1987)
 41. Lutz, C.: Reasoning with concrete domains. In: Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI 1999), Stockholm, pp. 90–95 (1999)
 42. Lutz, C.: The complexity of reasoning with concrete domains. Ph.D. thesis, Teaching and Research Area for Theoretical Computer Science, RWTH Aachen (2001)
 43. Lutz, C., Areces, C., Horrocks, I., Sattler, U.: Keys, nominals, and concrete domains. *J. Artif. Intell. Res.* 23, 667–726 (2004)
 44. Lutz, C., Toman, D., Wolter, F.: Conjunctive query answering in the description logic EL using a relational database system. In: Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009), Pasadena, pp. 2070–2075 (2009)
 45. MacGregor, R.: Inside the LOOM description classifier. *SIGART Bull.* 2(3), 88–92 (1991)

46. Morgan, A.P., Cafeo, J.A., Godden, K., Lesperance, R.M., Simon, A.M., McGuinness, D.L., Benedict, J.L.: The general motors variation-reduction adviser. *AI Mag.* **26**(2) (2005)
47. Motik, B.: On the properties of metamodeling in OWL. In: Proceedings of the Fourth International Semantic Web Conference (ISWC 2005), Galway. Lecture Notes in Computer Science, vol. 3729, pp. 548–562. Springer (2005)
48. Motik, B., Cuenca Grau, B., Horrocks, I., Sattler, U.: Representing structured objects using description graphs. In: Proceedings of the 11th International Conference on Principles of Knowledge Representation and Reasoning (KR 2008), Sydney, pp. 296–306 (2008)
49. Motik, B., Horrocks, I.: OWL datatypes: design and implementation. In: Proceedings of the Seventh International Semantic Web Conference (ISWC 2008), Karlsruhe. Lecture Notes in Computer Science, vol. 5318, pp. 307–322. Springer, Berlin (2008)
50. Motik, B., Sattler, U., Studer, R.: Query answering for OWL-DL with rules. In: Proceedings of the Third International Semantic Web Conference (ISWC 2004), Hiroshima. Lecture Notes in Computer Science, vol. 3298, pp. 549–563. Springer, Berlin (2004)
51. Motik, B., Shearer, R., Horrocks, I.: Optimized reasoning in description logics using hypertableaux. In: Proceedings of the 21st International Conference on Automated Deduction (CADE-21), Breman. Lecture Notes in Artificial Intelligence, vol. 4603, pp. 67–83. Springer, Berlin (2007)
52. OWL 2 web ontology language conformance, W3C Recommendation. <http://www.w3.org/TR/owl2-conformance/> (2009)
53. OWL 2 web ontology language direct semantics, W3C Recommendation. <http://www.w3.org/TR/owl2-direct-semantics/> (2009)
54. OWL 2 web ontology language RDF-based semantics, W3C Recommendation. <http://www.w3.org/TR/owl2-rdf-based-semantics/> (2009)
55. OWL 2 web ontology language Manchester syntax, W3C Working Group Note. <http://www.w3.org/TR/owl2-manchester-syntax/> (2009)
56. OWL 2 web ontology language overview, W3C Recommendation. <http://www.w3.org/TR/owl2-overview/> (2009)
57. OWL 2 web ontology language primer, W3C Recommendation. <http://www.w3.org/TR/owl2-primer/> (2009)
58. OWL 2 web ontology language profiles, W3C Recommendation. <http://www.w3.org/TR/owl2-profiles/> (2009)
59. OWL 2 web ontology language structural specification and functional-style syntax, W3C Recommendation. <http://www.w3.org/TR/owl2-syntax/> (2009)
60. Parsia, B., Sattler, U.: OWL 2 web ontology language data range extension: linear equations, W3C Working Group Note. <http://www.w3.org/TR/owl2-dr-linear/> (2009)
61. Patel-Schneider, P.F.: Small can be beautiful in knowledge representation. In: Proceedings of the IEEE Workshop on Principles of Knowledge-Based Systems, Denver (1984). An extended version appeared as Fairchild Technical Report 660 and FLAIR Technical Report 37
62. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL web ontology language semantics and abstract syntax, W3C Recommendation. <http://www.w3.org/TR/owl-semantics/> (2004)
63. Patel-Schneider, P.F., McGuinness, D.L., Brachman, R.J., Resnick, L.A., Borgida, A.: The CLASSIC knowledge representation system: guiding principles and implementation rational. *SIGART Bull.* **2**(3), 108–113 (1991)
64. Maryland Information and Network Dynamics Lab, Pellet OWL reasoner. <http://www.mindswap.org/2003/pellet/index.shtml> (2003)
65. Poggi, A., Lembo, D., Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R.: Linking data to ontologies. *J. Data Semant.* **10**, 133–173 (2008)
66. Internet Engineering Task Force (IETF): RFC 3987: Internationalized Resource Identifiers (IRIs). Request For Comments (RFC). <http://www.ietf.org/rfc/rfc3987.txt> (2005)
67. de Bruijn, J.: RIF RDF and OWL compatibility, W3C Recommendation. <http://www.w3.org/TR/rif-rdf-owl/> (June 2010)
68. Roberts, R.B., Goldstein, I.P.: The FRL primer. Memo 408, Artificial Intelligence Laboratory, Massachusetts Institute of Technology. <ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-408.pdf> (1977)
69. Robinson, A., Voronkov, A. (eds.): Handbook of Automated Reasoning. Elsevier, Amsterdam (2001)

70. Sattler, U.: A concept language extended with different kinds of transitive roles. In: Gorz, G., Holldobler, S. (eds.) *Proceedings of the 20th German Annual Conference on Artificial Intelligence (KI 1996)*, Dresden. *Lecture Notes in Artificial Intelligence*, vol. 1137, pp. 333–345. Springer, Berlin (1996)
71. Schild, K.: A correspondence theory for terminological logics: preliminary report. In: *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAF 1991)*, Sydney, pp. 466–471 (1991)
72. Schmidt-Schaufler, M.: Subsumption in KL-ONE is undecidable. In: Brachman, R.J., Levesque, H.J., Reiter, R. (eds.) *Proceedings of the First International Conference on the Principles of Knowledge Representation and Reasoning (KR 1989)*, Toronto, pp. 421–431. Morgan Kaufmann, Los Altos (1989)
73. Schmidt-Schaufler, M., Smolka, G.: Attributive concept descriptions with complements. *Artif. Intell.* **48**(1), 1–26 (1991)
74. Sirin, E., Parsia, B., Cuenca Grau, B., Kalyanpur, A., Katz, Y.: Pellet: a practical OWL-DL reasoner. *J. Web Semant.* **5**(2), 51–53 (2007)
75. Spackman, K.: Managing clinical terminology hierarchies using algorithmic calculation of subsumption: experience with SNOMED-RT. *J. Am. Med. Inform. Assoc.* (2000). Fall Symposium Special Issue
76. Spackman, K., Campbell, K., Cote, R.: SNOMED RT: a reference terminology for health care. *J. Am. Med. Inf. Assoc.* pp. 640–644 (1997). Fall Symposium Supplement
77. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF, W3C Recommendation. <http://www.w3.org/TR/rdf-sparql-query/> (Jan 2008)
78. Stevens, R., Baker, P., Bechhofer, S., Ng, G., Jacoby, A., Paton, N.W., Goble, C.A., Brass, A.: TAMBI: transparent access to multiple bioinformatics information sources. *Bioinformatics* **16**(2), 184–186 (2000)
79. ter Horst, H.J.: Completeness, decidability and complexity of entailment for RDF Schema and a semantic extension involving the OWL vocabulary. *J. Web Semant.* **3**(2–3), 79–115 (2005)
80. Tsarkov, D., Horrocks, I.: FaCT++ description logic reasoner: system description. In: *Proceedings of the Third International Joint Conference on Automated Reasoning (IJCAR 2006)*, Seattle. *Lecture Notes in Artificial Intelligence*, vol. 4130, pp. 292–297. Springer, Berlin (2006)

10 KR and Reasoning on the Semantic Web: RIF

Michael Kifer

State University of New York at Stony Brook, Stony Brook, NY, USA

10.1	<i>Introduction</i>	400
10.2	<i>Scenario: Procurement of Mobile Phone Services</i>	401
10.3	<i>Overview of RIF</i>	402
10.4	<i>The RIF Basic Logic Dialect</i>	405
10.4.1	Syntax	405
10.4.2	Datatypes	410
10.4.3	Semantics	411
10.4.4	Implementation Challenges	417
10.4.5	Built-ins and Semantic Puzzles	420
10.5	<i>RDF and OWL Compatibility</i>	422
10.5.1	Combination with RDF	423
10.5.2	Combination with OWL	425
10.6	<i>An Example Application in RIF</i>	427
10.7	<i>Conclusions</i>	436

Abstract: *Rule Interchange Format* (RIF) is a suite of W3C standards designed to facilitate rule exchange among different and dissimilar rule engines, especially among Web-enabled engines. Following on the heels of the earlier Semantic Web standards, RDF and OWL, RIF aims to revolutionize the field of Web application development and create infrastructure for truly intelligent Web applications. The goal of this chapter is to provide an overview of RIF, especially the syntax and semantics of its logic-based dialects. As an illustration, it is shown how RIF can be used to build a sophisticated distributed application for the procurement of mobile services, which heavily relies on rule-based reasoning. This chapter also discusses the limitations of RIF's *Basic Logic Dialect* (RIF-BLD); in particular, where it falls short of the requirements for complex applications, such as above, and shows how RIF's *Framework for Logic Dialects* (RIF-FLD) solves these problems by providing a general framework for designing more expressive dialects.

10.1 Introduction

Rule systems have been at the center of intense renewed interest in the past few years, as both industry and academia began to view rules as a key knowledge representation and reasoning technology – both in their own right and in combination with the existing Semantic Web standards, RDF [1] and OWL [2]. To help along with this trend, the World Wide Web Consortium (W3C) created a working group and chartered it with the task of designing a standardized language for exchanging rules among different and dissimilar rule engines – the *Rule Interchange Format* or RIF. After RDF and OWL, RIF is the latest installment in a series of Semantic Web standards. It is a suite of documents designed to facilitate rule exchange among systems, especially among Web-enabled rule engines. Web-enabled rule engines are engines that are aware of such semantic Web standards as IRIs [3], RDF [1], and can import and process distributed knowledge published as Web documents. Several key components of RIF have become W3C recommendations in June 2010. These documents as well as a number of related specifications are available on the RIF working group website [4]. It is envisaged that if significant uptake of the RIF standard will take place in the next few years, a powerful infrastructure for intelligent Web applications will emerge, and this will revolutionize the landscape of Web application development.

This chapter is intended to illustrate the exciting opportunities for distributed reasoning on the Web, which will arise once the RIF infrastructure is fully in place. First, an overview of RIF is given in order to help the reader navigate the RIF document suite. Next, a survey of RIF's *Basic Logic Dialect* (RIF-BLD) is presented together with the main ideas that underlie the combined RIF-RDF and RIF-OWL languages. Then this chapter develops a specific distributed application, procurement of mobile phone services, and shows how RIF can be used to put together such an application. For the most part, the example relies on RIF-BLD and illustrates, among other features, how one could take advantage of the repositories represented using other Semantic Web standards, such as RDF. At the same time, the example highlights the areas where RIF-BLD falls short of the needs of such sophisticated applications and, to overcome this handicap, certain features are borrowed from RIF-FLD,

the *RIF Framework for Logic Dialects*. RIF-FLD is not a dialect in itself, but a general logical framework that can be used for specifying dialects that extend RIF-BLD. The above example can thus be seen as an application of one of these future dialects.

This chapter is organized as follows. [▶ Section 10.2](#) introduces the service procurement example as a motivating use case. [▶ Section 10.3](#) introduces RIF and provides a general roadmap for this recently proposed standard. The next two sections, [▶ Sects. 10.4](#) and [▶ 10.5](#), review the details of RIF-BLD as well as of the main concepts underlying RIF compatibility with RDF and OWL. [▶ Section 10.6](#) develops the phone procurement scenario in detail using RIF as a specification language. [▶ Section 10.7](#) concludes the chapter.

On the technical side, this chapter assumes general familiarity with first-order logic syntax and semantics [5, 6] and with the idea of rule-based languages, especially logic programming languages like Prolog [7]. The reader who has prior knowledge of F-logic [8] and HiLog [9] will find the presentation of the syntax and semantics more familiar.

10.2 Scenario: Procurement of Mobile Phone Services

To illustrate how RIF is envisaged to provide a reasoning infrastructure for the Semantic Web, consider the following example of a Web-based system for the procurement of mobile phone services.

The main players in the business of mobile service procurement are service providers (cell phone companies), cell phone manufacturers, and customers (end users of phone services). After learning about this latest cool technology called RIF, a group of entrepreneurial teenage hackers commandeers a family garage and puts together an aggregator website that collects and organizes information obtained from service providers and manufacturers, combines this information with various FOAF relationships [10] and user profiles, and lets customers make informed decisions about the most suitable services and phones that fit the requirements.

When shopping for a new mobile phone service, a customer typically has a number of soft and hard requirements. For instance, the customer might prefer GSM or a CDMA type of service, he or she might want a smartphone or this might be unimportant, Wi-Fi might be another requirement as well as a touch screen. In addition, customers might want prepaid or monthly plans, the most amount of minutes for the cheapest price. Furthermore, since many providers throw in free or discounted minutes for calls made within the same network, a FOAF ontology might help prospective customers to choose the service that is the cheapest overall by taking into account the mobile providers used by their closest friends.

To help customers with their choices, aggregator sites integrate phone ontologies, which provide information about the various phones on the market, with information supplied by the phone service providers. The latter includes description of the various plans, discounts, and the phones that go with each plan. Some of this information may be stored in OWL and RDF ontologies, while other information might live in various RIF documents. Together with the FOAF ontology, this creates a fairly complex environment where information is drawn from distributed ontologies, RIF knowledge bases, and all of

these are glued together with RIF rules. This scenario is formally developed in [Sect. 10.6](#), but first one needs to become familiar with RIF.

10.3 Overview of RIF

The Rule Interchange Format (RIF) working group was chartered by the World Wide Web Consortium (W3C) in 2005 to create a standard for exchanging rules among rule systems in general, but especially among Web-based rule engines. W3C went the exchange route rather than trying to develop a single one-size-fits-all rule language because, unlike in other similar cases (XQuery [11], SPARQL [12], OWL [2]), it was immediately clear that a single language will not cover all the popular paradigms of using rules for knowledge representation and business processes. Even rule exchange alone was recognized as a daunting task. Known rule systems fall into three general categories: logic programming, first-order rules, and action rules. These paradigms share very little in terms of syntax and semantics, and there are huge differences between systems even within the same paradigms.

Given this diversity, can there be any useful notion of exchange? The approach taken by the working group was to design a family of languages, called *dialects*, with a rigorously specified syntax and semantics. The family of RIF dialects was intended to be *extensible* and *uniform*. Extensibility here means that it should be possible to add new dialects that various user groups might want to develop. RIF uniformity means that dialects are expected to share much of the syntactic and semantic apparatus.

Because of the emphasis on rigor, the word “format” in the name of RIF might seem like a misnomer. To clarify, RIF is a format in the sense that ultimately the medium of exchange is XML – a format for data exchange. The main idea behind rule exchange through RIF is that the different systems will provide syntactic mappings from their native languages to RIF dialects and back. These mappings are required to be *semantics-preserving* and thus rule sets and data could be communicated by one system to another provided that the systems can talk through a suitable dialect, which they both support.

RIF Dialects. The RIF Working Group has been focusing on two kinds of dialects: *logic-based dialects* and dialects for *rules with actions*. Generally, logic-based dialects include languages that employ some kind of a logic, such as the first-order logic or non-first-order logics, underlying the various logic programming languages (e.g., logic programming under the well-founded or stable semantics [13, 14]). The rules-with-actions dialects include production rule systems, such as Jess, Drools [15, 16], and JRules [17], and reactive (or ECA) rules, such as XChange [18] and Prova [19]. Given the limited resources of the RIF working group, it defined only two logic dialects: the *Core* dialect (RIF-Core) and its extension the Basic Logic Dialect (RIF-BLD) [20]; and only one rules-with-actions dialect: the *Production Rule Dialect* (RIF-PRD) [21]. Other dialects are expected to be defined by the various user communities. [Section 10.4](#) discusses RIF-BLD in detail.

RIF Framework for Logic Dialects. The RIF working group spent almost 4 years on developing the aforesaid three dialects, and this begs a question: If dialect development is so time consuming, who will donate the necessary resources for the next round of

development and who will ensure the uniformity of community-developed dialects once the RIF working group disbands? The working group partially addressed these questions by developing an extensibility framework, called the Framework for Logic Dialects, or RIF-FLD [22]. A comparable framework for rules with actions is not likely to be developed by the RIF working group, but it might be developed later by other interest groups. For example, a framework that captures much of the reactive rules paradigm, as it exists in XChange [18] and Prova [19] as well as in rule systems like FLORA-2 [23] and SILK [24], can be developed as an extension to RIF-FLD.

Development of the RIF framework turned out to be feasible because despite the diversity of logical theories underlying the different logic rule systems, they share much of the syntactic and semantic machinery, and the ways to combine the different pieces of that machinery in order to create those systems are well-studied. However, the RIF-FLD specification [22] is unique in that it digests much of this knowledge and presents it in a coherent form.

RIF-FLD is a very general logic language that includes a great deal of commonly used syntactic and semantic gadgetry, but it purposely leaves certain parameters unspecified to enable the designers of the concrete dialects fill in the necessary details. For instance, RIF-FLD provides machinery to tweak the rules of syntax through the notion of *signatures*. It also specifies certain semantic notions, such as models and logical entailment, but it leaves certain other options open (for instance, which exact models are to be used for entailment). A dialect designer can then define the syntax of a dialect by *specializing* it from the syntax of RIF-FLD and the semantics by specializing it from the semantics of RIF-FLD. While doing so, the designer makes choices by selecting the options provided by RIF-FLD, but it does not have to repeat the definitions of formulas, datatypes, models, entailment, and so on. This approach is illustrated using the RIF-BLD dialect [20]. This dialect is specified in two ways, both normative: directly, by spelling out all the definitions, which takes about 40 dense pages, and by specialization from RIF-FLD – just about five pages. Any discrepancy between the two specifications is supposed to be treated as a “bug” that must be clarified and corrected. This dual specification of RIF-BLD is also intended to serve as an example of dialect design by specialization from the RIF framework – the preferred mode of specification for various future logical dialects. A more detailed overview of RIF-FLD can be found in [25].

The RIF framework is not a monument that is cut in stone and is likely to see several extensions in the future. One, as already mentioned, might be to cover the paradigm of reactive rules.

It should also be noted that the dialects derived from RIF-FLD are purely logical and, therefore, they do not cover Prolog. For instance, RIF-Core and RIF-BLD capture only a small subset of Prolog, one which can be considered purely logical. However, since Prolog’s proof strategy is incomplete even for its purely logical subsets, these subsets cannot be faithfully exchanged through the logical RIF dialects.

RDF and OWL compatibility. Recognizing that RIF rules should be able to interface with RDF and OWL ontologies, the working group also defined the necessary concepts to ensure compatibility of RIF with RDF and OWL. RIF, RDF, and OWL are exchange

languages with dissimilar syntaxes and semantics. How, then, should RIF rules refer to RDF and OWL facts, and what is the logical meaning of the overall language? The RIF-RDF and OWL compatibility document [26] defines just that. The basic idea is that RIF uses its frame syntax to communicate with RDF/OWL. These frames are mapped onto RDF triples and joint semantics is defined for the combination. This is explored in more detail in [Sect. 10.5](#).

A guide to RIF documents. The RIF working group has produced a significant number of documents (see [4]), some of which have already become W3C recommendations (i.e., standards). Here is a brief description of the key documents.

- *RIF-BLD: The Basic Logic Dialect* [20]. As already discussed in this section, this is one of the two main dialects developed by the group so far and the main logic-based dialect. Technically, this dialect corresponds to Horn logic with various syntactic and semantic extensions. The main syntactic extensions include the frame syntax and predicates with named arguments. The main semantic extensions include datatypes and externally defined predicates. Although this dialect is not expressive enough for many interesting applications (including the one in [Sect. 10.6](#)), it covers many of the existing rule systems, and development of such a dialect was necessary as a starting point for the future, more expressive dialects. This future activity is expected to take place within the RIF extensibility framework, RIF-FLD.
- *RIF-PRD: The Production Rule Dialect* [21]. This dialect tries to capture the main aspects of the various production rule systems. Serious industrial interests are lined up behind the production rules technology now, with major players including IBM (e.g., IBM JRules) and Oracle (e.g., Oracle Business Rules). Production rules, as they are currently practiced in mainstream systems like Jess [15] or JRules [17], are defined using ad hoc computational mechanisms, which have little to do with logic. For this reason, RIF-PRD is not part of the suite of logical RIF dialects and stands apart from them. However, significant effort has been extended to ensure as much sharing with the other dialects as possible. This sharing was the main reason for the development of the RIF Core dialect.
- *RIF-Core: The Core Dialect* [27]. This dialect is a subset of both RIF-BLD and RIF-PRD. By itself, RIF-Core is a rather inexpressive dialect. Its main purpose is to enable limited rule exchange between logic rule dialects and production rules.
- *RIF-FLD: The Framework for Logic Dialects* [22]. As discussed earlier in this section, RIF-FLD is not a dialect in its own right, but rather a general logical extensibility framework. It was introduced in order to drastically lower the amount of effort needed to define and verify new logic dialects that extend the capabilities of RIF-BLD. RIF-FLD makes it possible to define very expressive rule dialects in a way that preserves the semantic and syntactic unity needed for rule exchange. Apart from RIF-BLD, examples of the dialects that have already been defined using RIF-FLD include rules with well-founded semantics (<http://ruleml.org/rif/RIF-CLPWD.html>), answer-set semantics (<http://ruleml.org/rif/RIF-CASPD.html>), and a dialect that supports reasoning with uncertainty (http://ruleml.org/rif/URSW2008_F9_ZhaoBoley.pdf).

- *RIF-RDF+OWL: RDF and OWL Compatibility* [26]. This document enables interoperability between RIF and other Semantic Web standards: RDF and OWL. It defines the syntax and semantics of combined RIF+RDF and RIF+OWL 2 languages.
- *RIF-DTB: Datatypes and Built-ins* [28]. Rules often refer to built-ins (e.g., arithmetics, string manipulation) and datatypes (e.g., integers, strings, Booleans). To enable the semantics-preserving exchange of such rules, it is necessary that most of the commonly used datatypes and built-in functions are identified and their semantics defined precisely. This purpose is served by the RIF-DTB document.
- *RIF-UCR: Use Cases and Requirements* [29]. One of the first tasks of the RIF working group was to identify classes of applications that the RIF suite of dialects should be able to address, and use that to derive requirements for RIF. To a large extent, the design of RIF dialects was driven by the requirements found in the RIF-UCR document.
- *RIF-Test: Test Cases* [30]. This document is primarily of concern to RIF implementers. It includes the description of test cases – both positive and negative – that can be used in order to give an indication of whether a particular implementation of a RIF dialect is compliant with the specifications. There is a companion repository of the source code for the various test cases [30].

Additional RIF documents describe ways of integrating XML data sources with RIF (http://www.w3.org/2005/rules/wiki/RIF+XML_data-schema) and a representation of the rule profile of OWL 2 (OWL 2 RL) in RIF-Core (<http://www.w3.org/2005/rules/wiki/OWLRL>). Another document, RIF Primer (<http://www.w3.org/2005/rules/wiki/Primer>), provides a gentle introduction to the main concepts of RIF.

10.4 The RIF Basic Logic Dialect

The Basic Logic Dialect, RIF-BLD, is a rule language that is essentially Horn logic extended with datatypes and a number of syntactic features, notably the frame syntax borrowed from F-logic [8, 31]. This section provides a brief overview of the syntax and semantics of RIF-BLD.

10.4.1 Syntax

RIF dialects normally have two syntaxes: an *XML-based syntax* and a *presentation syntax*. The former is a normative syntax – one that is actually used for rule exchange and is understood by RIF processors. However, XML is insufficient as a sole means of language specification for RIF, as RIF dialects are complex languages that cannot be completely defined using XML alone. First, logic languages are typically not context-free and

thus cannot be specified using DTDs [32] or XML Schema [33]. Second, XML is too verbose to be a serious contender when it comes to specifying the semantics of a logic language. Thus, a more traditional presentation syntax is used for specifying the semantics of RIF-BLD and other dialects. Since the semantics is normative, so is the presentation syntax. The presentation syntax is also used for human-readable examples and use cases. However, it should be kept in mind that the presentation syntax is an *abstract syntax* – it describes parse trees, but it is not sufficiently concrete to enable parsing. Many things, including the precise specification of delimiters, have deliberately been left out.

As mentioned earlier, RIF-BLD is defined in two ways: *indirectly*, as a specialization of the RIF logical framework, RIF-FLD [22], and also *directly*, from scratch. The first method is very short; it is intended for the reader who is familiar with RIF-FLD, primarily a new dialect designer or a user of multiple RIF dialects. The direct specification is for people whose interests are confined to the basic dialect.

To make this chapter self-contained, the direct specification method is chosen. The XML syntax is not discussed at all, since the purpose of this chapter is to introduce the reader to the idea of Web-based distributed reasoning with RIF – it is not a RIF manual.

Definition 1 (Alphabet). The *alphabet* of the presentation language of RIF-BLD consists of the following disjoint sets of symbols:

- A countably infinite set of *constant symbols* Const
- A countably infinite set of *variable symbols* Var
- Connective symbols And , Or , and :
- Various auxiliary symbols, such as $\#$, -> , = , parentheses, and so on.

Variables are written as Unicode strings preceded with the symbol “?”. Constants are written as `"literal"^^symspace`, where `literal` is a sequence of Unicode characters and `symspace` is an identifier for a symbol space (see below). The symbols = , $\#$, and $\#\#$ are used to represent equality, class membership, and subclass relationships. The symbol -> is used in frame formulas. Additional symbols, such as `External`, indicate that an atomic formula or a function term is defined externally (e.g., a built-in); the symbols `Prefix` and `Base` are used for abridged representation of IRIs [3].

Other auxiliary symbols include `Document` – for specifying RIF-BLD documents, `Import` – for importing documents, and `Group` – for organizing RIF-BLD formulas into collections. □

Symbol spaces mentioned in the above definitions are subsets that partition the space of all constants Const .

Definition 2 (Symbol spaces). A *symbol space* has an *identifier* and a *lexical space* that defines the “shape” (syntax) of the symbols in that symbol space. For example, the symbol space of integers has an identifier <http://www.w3.org/2001/XMLSchema#integer>, and its lexical space consists of character strings that look like integer numbers.

Some symbol spaces in RIF are used as identifiers for Web entities, and their lexical space consists of strings that syntactically look like *internationalized resource identifiers*, or *IRIs* [3] (e.g., <http://www.w3.org/2007/rif#iri>). Most of the other symbol spaces are used to represent the datatypes required by RIF. For example,

```
http://www.w3.org/2001/XMLSchema#integer
http://www.w3.org/2001/XMLSchema#decimal
http://www.w3.org/2001/XMLSchema#boolean
http://www.w3.org/2001/XMLSchema#dateTime
http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral
```

Another important symbol space in RIF is `rif:local`. The `rif:local` constants are used to name function and predicate symbols that are local to a particular document and are not accessible from outside of that document (i.e., they are “encapsulated” within that document).

Every constant in `Const` belongs to exactly one symbol space. □

Note that although each constant symbol belongs to a unique symbol space, this does not preclude equality among constants that belong to different symbol spaces. For instance, the constants

```
"1"^^<http://www.w3.org/2001/XMLSchema#integer>
"1"^^<http://www.w3.org/2001/XMLSchema#decimal>
```

are equal according to the RIF semantics, that is,

```
"1"^^<http://www.w3.org/2001/XMLSchema#integer>
  = "1"^^<http://www.w3.org/2001/XMLSchema#decimal>
```

is a true statement.

RIF-BLD formulas are constructed according to the rules of syntax described next.

Terms. The main building blocks in the RIF-BLD language are *terms*. RIF-BLD defines several kinds of terms: *constants* and *variables*, *positional* and *named-argument* terms, *lists*, *equality*, *membership*, *subclass*, *frame*, and *external* terms. This exposition, omits some types of terms, such as lists and named-argument terms, in order to focus on the core functionality.

Definition 3 (Term). The notion of a term is defined by structural recursion. In this definition, *base term* refers to simple, positional, or terms of the form `External(t)`.

1. *Constants and variables.* If $t \in \text{Const}$ or $t \in \text{Var}$, then t is a simple term.
2. *Positional terms.* If $t \in \text{Const}$ and $t_1, \dots, t_n, n \geq 0$, are base terms, then $t(t_1 \dots t_n)$ is a positional term.

Positional terms correspond to the usual terms and atomic formulas of classical first-order logic.

3. *Equality terms.* $t = s$ is an equality term, if t and s are base terms.
4. *Class membership terms* (or just *membership terms*). $t \# s$ is a membership term if t and s are base terms.
5. *Subclass terms.* $t \# \# s$ is a subclass term if t and s are base terms.
6. *Frame terms.* $t [p_1 \rightarrow v_1 \dots p_n \rightarrow v_n]$ is a frame term (or simply a *frame*) if $t, p_1, \dots, p_n, v_1, \dots, v_n, n \geq 0$, are base terms.
7. *Externally defined terms.* If t is a positional term, then $\text{External}(t)$ is an externally defined term.

Membership, subclass, and frame terms are used to describe objects and class hierarchies in object-oriented logic languages such as F-logic [8].

External terms are used for representing built-in functions and predicates as well as “procedurally attached” terms or other predicates that are specified outside of RIF.

Immediate nesting of external terms, as in $\text{External}(\text{External}(\dots))$, is not allowed, but external terms *can* otherwise be nested, as for instance in $\text{External}(p(\text{External}(q(\dots)), ?X))$. This shows an external function, p , with an embedded external function q . \square

Observe that the argument names of frame terms, p_1, \dots, p_n , are base terms and so, as a special case, can be variables or, more generally, positional terms that contain variables.

Formulas. RIF-BLD distinguishes certain subsets of Const , including the subset of *predicate symbols* and *function symbols*. It further separates external function and predicate symbols (like p in $\text{External}(p(\dots))$) from the regular function/predicate symbols. This separation is a nod toward the standard first-order syntax used in OWL, but it is not really necessary. Future dialects that will extend RIF-BLD in various directions are likely to drop this restriction.

Any term of the form $p(\dots)$, where p is a predicate symbol, is also an *atomic formula*. Equality, membership, subclass, and frame terms are also atomic formulas. An externally defined term of the form $\text{External}(\varphi)$, where φ is an atomic formula, is also an atomic formula, called an *externally defined* atomic formula. More general formulas are constructed with the help of logical connectives.

Definition 4 (Formula). A *formula* can have several different forms:

1. *Atomic.* If φ is an atomic formula, then it is also a formula.
2. *Condition formula.* A *condition formula* is either
 - An atomic formula
 - A *conjunctive* formula $\text{And}(\varphi_1 \dots \varphi_n)$, where $\varphi_1, \dots, \varphi_n$ are condition formulas
 - A *disjunctive* formula $\text{Or}(\varphi_1 \dots \varphi_n)$, where $\varphi_1, \dots, \varphi_n$ are as above or
 - An *existential* formula $\text{Exists } ?V_1 \dots ?V_n(\varphi)$, where φ is a condition formula and $?V_1, \dots, ?V_n$ are distinct variables.

Condition formulas are intended to be used inside rule premises.

3. *Rule implication.* $\varphi :- \psi$ is a formula, called *rule implication*, if:

- φ is an atomic formula or a *conjunction* of atomic formulas, it is the *rule conclusion*
- ψ is a condition formula, it is the *rule premise*
- None of the atomic formulas in φ is an externally defined term (i.e., a term of the form `External(...)`)

External terms *are* allowed in the *arguments* to atomic formulas in the rule conclusion, but they cannot occur as atomic formulas. For instance,

```
External(p(...)) :- premises
```

is *not* allowed, but

```
q(External(p(...))) :- premises
```

is a valid rule implication in which `p` is an external function symbol. Equality is also allowed in rule conclusions. For instance,

```
?X = ?Y :- And(?X[url->?XU]
               ?Y[url->?YU]
               External(same_url(?XU,?YU)))
```

This rule says that if two syntactically distinct IRI constants point to the same page (assuming that this is what the external predicate `same_url` means), then these two constants must be deemed logically equal.

4. *Universal rule.* If φ is a rule implication and $?V_1, \dots, ?V_n, n > 0$, are distinct variables then

```
forall ?V_1 ... ?V_n(\varphi)
```

is a *universal rule* formula. Universal rules are also called *RIF-BLD rules*. Unlike logic programming, quantification is mandatory in RIF-BLD, but this requirement is disregarded in the examples, for readability. A *universal fact* is a universal rule with no premises.

5. *Group.* If $\varphi_1, \dots, \varphi_n$ are RIF-BLD rules, universal facts, variable-free rule implications, variable-free atomic formulas, or group formulas then

```
Group(\varphi_1 ... \varphi_n)
```

is a *group formula*. Note that some of the φ_i 's can be group formulas themselves, so groups can be nested.

Group formulas are used to represent sets of rules and facts, and can serve as attachment points for meta-information or for other structuring purposes. Representation of meta-information in RIF is not discussed in this chapter. It should be mentioned, however, that such meta-information can include identification, authorship, and much more.

6. *Document:* An expression of the form

```
Document(directive_1 ... directive_n \Gamma)
```

is a *RIF-BLD document formula* (or simply a *document formula*), if

- Γ is an optional group formula; it is called the group formula *associated* with the document.
- *directive*₁, . . . , *directive*_n is an optional sequence of **directives**. A directive can be a *base directive*, a *prefix directive*, or an *import directive* to the others.

- The base directive has the form `Base(<v>)`, where `v` is a string that forms an IRI. Whenever a relative IRI is used in a RIF document, the base directive specifies the missing absolute part. For example, if `Base(<http://example.com/>)` is specified and the document contains an IRI such as `"foo"^^rif:iri` then RIF processors are supposed to interpret this IRI as `"http://example.com/foo"^^rif:iri`.
- A prefix directive has the form `Prefix(p <v>)`, where `p` is an alphanumeric string that serves as the prefix name and `v` is an expansion for `p` – a string that forms an IRI.

The `Prefix` directives define shorthands to enable the compact URI representation for constants that come from the symbol space `rif:iri` (these are called *rif:iri constants*). For instance, with a prefix directive `Prefix(ex <http://example.com/>)`, the `rif:iri` constant `"http://example.com/foobar"^^rif:iri` can be written more succinctly as `ex:foobar`.

The compact URI notation will be frequently used throughout this chapter.

- An `import` directive has the form `Import(<t>)` or `Import(<t> <p>)`. Here `t` and `p` are Unicode sequences of characters that have the form of an IRI. The constant `t` indicates the location of another document to be imported, and `p` is called the *profile of import*. [▶ Section 10.4.3](#) defines the semantics for the directive `Import(t)` only. The two-argument directive, `Import(tp)`, is intended for importing non-RIF-BLD documents, such as rules from other RIF dialects, RDF data, or OWL ontologies. The profile, `p`, indicates what kind of entity is being imported and under what semantics (for instance, the various RDF entailment regimes have different profiles). The semantics of `Import(tp)` (for various `p`) are expected to be given by other specifications on a case-by-case basis. As an example, [26] defines the semantics for the profiles that are recommended for importing RDF and OWL – see [▶ Sect. 10.5](#). □

Note that the definition of RIF-BLD rules does not allow any kind of negation in rule premises – neither classical [5] nor negation as failure in its many forms [13, 14, 34]. So, RIF-BLD rules are Horn [35]. This is precisely why RIF-BLD is called the basic logic dialect: Horn rules is the bifurcation point between classical-logic-based rules and other kinds of rules (production, logic programming) [35].

10.4.2 Datatypes

Datatypes play an important role in the overall RIF infrastructure. Formally, they are defined as follows:

Definition 5 (Datatype). *A datatype is a symbol space that, in addition, has the following components:*

- An associated set, called the value space
- A mapping from the lexical space of the symbol space to the value space, called lexical-to-value-space mapping. □

The semantics of RIF dialects is always defined with respect to a particular set of datatypes, denoted by *DTS*. In a concrete dialect, *DTS* always includes the datatypes supported by that dialect. All RIF dialects must support the datatypes that are listed in Section “Datatypes” of [28]. The value spaces and the lexical-to-value-space mappings for these datatypes are described in the same document.

While listing all the datatypes, their value spaces, and mappings is tedious and not particularly elucidating, looking at a few concrete examples could be instructive. For the `xs:integer` (i.e., <http://www.w3.org/2001/XMLSchema#integer>) datatype, the value space is the set of all “real life” integers. The lexical-to-value-space mapping maps the lexical element “1” of the symbol “1”^{xs:integer} to the actual integer 1, the lexical element “-1” to the actual integer -1, and so on. In this case, the mapping is one-to-one. For the datatype `xs:decimal` (i.e., <http://www.w3.org/2001/XMLSchema#decimal>), the value space is the set of all rational numbers. But the lexical-to-value-space is no longer one-to-one. Indeed, the symbols “1.1”^{xs:decimal} and “1.100”^{xs:decimal} are distinct and their lexical parts (“1.1” and “1.100”) are also distinct. However, they represent the same rational number and thus are mapped to the same element in the value space of that datatype, that is, the equality “1.1”^{xs:decimal} = “1.100”^{xs:decimal} holds.

In sum, although the lexical and the value spaces might sometimes look similar, one should not confuse them. Lexical spaces define the *syntax* of the constant symbols in the RIF language. Value spaces define the *meaning* of the constants. As the above example of decimals shows, the lexical and the value spaces are often not even isomorphic.

10.4.3 Semantics

This section is an overview of the main aspects of the semantics of RIF-BLD. As before, the direct specification method is chosen, which does not require familiarity with RIF-FLD. While going through this direct specification, the reader might notice that it is considerably more general than what is strictly required for such a relatively simple rule language as the Basic Logic Dialect. The official specification [20] is written this way in order to ensure that the semantics of RIF-BLD does not diverge from RIF-FLD and that the planned future RIF logic dialects will be proper extensions of the Basic Logic Dialect. As the working versions of the documents keep changing, this approach also simplifies the process of maintaining the equivalence between the directly specified semantics of RIF-BLD and the semantics obtained by specialization from RIF-FLD.

The definition of the semantics employs only the full syntax of RIF-BLD, as it is assumed that the various shortcuts permitted by the syntax (such as compact URIs) have already been expanded into their full form. Such an expansion can be performed by a simple preprocessor.

Semantic Structures. The key concept in a model-theoretic semantics of a logic language is the notion of *semantic structures* [5]. In RIF-BLD, it is an adaptation of the standard semantics for Horn clauses, although it uses general domains instead of Herbrand domains. The decision to rely on general semantic structures rather than the Herbrand ones was based on the anticipation of future dialects that might extend RIF-BLD in the direction of full first-order logic or some expressive subset of it.

Definition 6 (Semantic structure). A *semantic structure*, \mathcal{I} , is a tuple of the form $\langle TV, DTS, D, D_{\text{ind}}, D_{\text{func}}, I_C, I_V, I_F, I_{\text{frame}}, I_{\text{sub}}, I_{\text{isa}}, I_{=} , I_{\text{external}}, I_{\text{truth}} \rangle$. Here TV denotes the set of truth values $\{\mathbf{t}, \mathbf{f}\}$. D is a nonempty set of elements, called the *domain* of \mathcal{I} , and $D_{\text{ind}}, D_{\text{func}}$ are nonempty subsets of D . D_{ind} is used to interpret the elements of Const that play the role of individuals and D_{func} is used to interpret the constants that play the role of function symbols. As before, Const denotes the set of all constant symbols and Var the set of all variable symbols. DTS denotes a set of identifiers for primitive datatypes.

The remaining components of \mathcal{I} are *total* mappings defined as follows:

1. I_C is a mapping $\text{Const} \rightarrow D$.
This mapping interprets constant symbols. In addition, it is required that:
 - If $c \in \text{Const}$, is an *individual* constant symbol, then $I_C(c) \in D_{\text{ind}}$. (Recall that the language of RIF-BLD distinguishes individual constant symbols from the symbols used to denote predicates and functions.)
 - If $c \in \text{Const}$, is a function symbol, then $I_C(c) \in D_{\text{func}}$.
2. I_V is a mapping $\text{Var} \rightarrow D_{\text{ind}}$.
It interprets variable symbols as individuals.
3. I_F maps D to total functions $D_{\text{ind}}^* \rightarrow D$ (here D_{ind}^* is a set of all finite sequences over the domain D_{ind}).

This mapping interprets function terms. In addition:

- If $d \in D_{\text{func}}$, then $I_F(d)$ must be a function $D_{\text{ind}}^* \rightarrow D_{\text{ind}}$. This implies that when a function symbol is applied to arguments that are individual objects, then the result is also an individual object.
4. I_{frame} maps D_{ind} to total functions of the form $\text{SetOfFiniteBags}(D_{\text{ind}} \times D_{\text{ind}}) \rightarrow D$

This mapping interprets frame terms. An argument $d \in D_{\text{ind}}$ to I_{frame} represents an object, and $\{ \langle a_1, v_1 \rangle, \dots, \langle a_k, v_k \rangle \}$ is a finite bag (multiset) of attribute-value pairs for d . It will be shown shortly how I_{frame} is used to determine the truth valuation of frame terms.

Bags are used here because the order of the attribute-value pairs in a frame is immaterial and pairs may repeat: $\circ [a \rightarrow b \ a \rightarrow b]$. Such repetitions arise naturally when variables are instantiated with constants. For instance, $\circ [?A \rightarrow ?B \ ?C \rightarrow ?D]$ becomes $\circ [a \rightarrow b \ a \rightarrow b]$ if variables $?A$ and $?C$ are instantiated with the symbol a and $?B, ?D$ with b . (It will be seen later that $\circ [a \rightarrow b \ a \rightarrow b]$ and $\circ [a \rightarrow b]$ are actually equivalent.)

5. I_{sub} gives meaning to the subclass relationship. It is a mapping of the form

$$D_{\text{ind}} \times D_{\text{ind}} \rightarrow D$$

An additional restriction, given in [Sect. 10.4.3](#), ensures that the operator $\#\#$ is transitive, that is, that $c1 \#\# c2$ and $c2 \#\# c3$ imply $c1 \#\# c3$.

6. I_{isa} gives meaning to class membership. It is a mapping of the form

$$D_{\text{ind}} \times D_{\text{ind}} \rightarrow D$$

An additional restriction, specified in [Sect. 10.4.3](#), ensures that the relationships $\#$ and $\#\#$ have the usual property that all members of a subclass are also members of the superclass, that is, that $o \# c1$ and $c1 \#\# sc1$ imply $o \# sc1$.

7. $I_{=}$ is a mapping of the form $D_{\text{ind}} \times D_{\text{ind}} \rightarrow D$.

It gives meaning to the equality operator.

8. I_{truth} is a mapping of the form $D \rightarrow TV$.

It is used to define truth valuation for formulas.

9. I_{external} is a mapping that gives meaning to `External` terms. It maps symbols in `Const` designated as external to fixed functions of appropriate arity. (Certain details are omitted for simplicity; see [20] for a full exposition.) Typically, external terms are invocations of built-in functions or calls to external non-RIF sources, and their fixed interpretations are determined by the specification of those built-ins and external sources.

The following mapping from terms to D , which is denoted by the same symbol I as the one used for semantic structures, is used in defining the truth value of a formula.

- $I(k) = Ic(k)$, if k is a symbol in `Const`
- $I(?v) = Iv(?v)$, if $?v$ is a variable in `Var`
- $I(f(t_1 \dots t_n)) = If(I(f))(I(t_1), \dots, I(t_n))$
- $I(o[a_1->v_1 \dots a_k->v_k]) = I_{\text{frame}}(I(o))(\{<I(a_1), I(v_1)>, \dots, <I(a_n), I(v_n)>\})$

Here $\{. . .\}$ denotes a bag of attribute-value pairs. The semantics is defined in such a way that duplicate elements in the above bag do not affect the truth value of a frame formula. So, for instance, $o[a->b a->b]$ and $o[a->b]$ always have the same truth value.

- $I(c1\#\#c2) = I_{\text{sub}}(I(c1), I(c2))$
- $I(o\#c) = I_{\text{isa}}(I(o), I(c))$
- $I(x=y) = I_{=}(I(x), I(y))$
- $I(\text{External}(p(s_1 \dots s_n))) = I_{\text{external}}(p)(I(s_1), \dots, I(s_n))$.

Interpretation of datatypes. In addition, RIF-BLD imposes certain restrictions on datatypes in DTS so that they are interpreted as intended: for instance, the constants in the symbol space `xsd:integer` are interpreted by integers.

More formally, the datatype identifiers mentioned in DTS must satisfy the following restrictions. If $dt \in DTS$, let LS_{dt} be the lexical space of dt , VS_{dt} be its value space, and $L_{dt} : LS_{dt} \rightarrow VS_{dt}$ be the associated lexical-to-value-space mapping, then the following must hold:

- $VS_{dt} \subseteq D$
- For each constant, "lit"^^dt such that $lit \in LS_{dt}$ $I_C("lit"^^dt) = L_{dt}(lit)$.

That is, I_C maps the constants of a datatype dt as prescribed by the lexical-to-value-space mapping L_{dt} . \square

Note how this definition implies various equalities among constants from different symbol spaces. For instance, according to the XML Datatypes specification [36], the value space of the datatype of integers is a subset of the value space of decimals. This implies various equalities such as

```
"123"^^<http://www.w3.org/2001/XMLSchema#integer>
  = "123"^^<http://www.w3.org/2001/XMLSchema#decimal>
```

Interpretation of Non-document Formulas. First, consider RIF-BLD formulas *other than* document formulas. Their truth valuation is defined using a mapping, $TVal_{\mathcal{I}}$, from the set of all non-document formulas to \mathbf{TV} .

Observe that in the case of atomic formulas, $TVal_{\mathcal{I}}(\phi)$ is defined essentially as $I_{\text{truth}}(\mathbf{I}(\phi))$. Recall that $\mathbf{I}(\phi)$ is just an element of the domain \mathbf{D} and I_{truth} maps \mathbf{D} to truth values in \mathbf{TV} . This might seem unusual to readers who are used to standard textbook-style definitions [5, 6], since normally the mapping \mathbf{I} is defined only for terms that occur as arguments to predicates, not for atomic formulas. Likewise, traditionally truth valuations have been defined via mappings from instantiated formulas to \mathbf{TV} , not from the interpretation domain \mathbf{D} to \mathbf{TV} . The style of definition used in RIF-BLD is inherited from RIF-FLD [22] and is equivalent to a standard one for a first-order language such as RIF-BLD. Originally, this style was introduced in HiLog [9] and was adopted by RIF-FLD to support future RIF dialects that include higher-order language features and reification (e.g., HiLog [9], FLORA-2 [23, 37]).

Definition 7 (Truth valuation). *Truth valuation* for well-formed formulas in RIF-BLD is determined using the function $TVal_{\mathcal{I}}$ as follows:

1. *Atomic formulas:* $TVal_{\mathcal{I}}(x(t_1 \dots t_n)) = I_{\text{truth}}(\mathbf{I}(x(t_1 \dots t_n)))$
2. *Equality:* $TVal_{\mathcal{I}}(x = y) = I_{\text{truth}}(\mathbf{I}(x = y))$.
 - To ensure that equality has precisely the expected properties, it is required that:
 - $I_{\text{truth}}(\mathbf{I}(x = y)) = \mathbf{t}$ if $\mathbf{I}(x) = \mathbf{I}(y)$
 - $I_{\text{truth}}(\mathbf{I}(x = y)) = \mathbf{f}$ otherwise. This is tantamount to saying that $TVal_{\mathcal{I}}(x = y) = \mathbf{t}$ if and only if $\mathbf{I}(x) = \mathbf{I}(y)$.
3. *Subclass:* $TVal_{\mathcal{I}}(sc \## c1) = I_{\text{truth}}(\mathbf{I}(sc \## c1))$.
To ensure that the operator $\##$ is transitive, that is, $c1 \## c2$ and $c2 \## c3$ imply $c1 \## c3$, the following is required:
 - For all $c1, c2, c3 \in \mathbf{D}$, if $TVal_{\mathcal{I}}(c1 \## c2) = TVal_{\mathcal{I}}(c2 \## c3) = \mathbf{t}$ then $TVal_{\mathcal{I}}(c1 \## c3) = \mathbf{t}$.
4. *Membership:* $TVal_{\mathcal{I}}(o \# c1) = I_{\text{truth}}(\mathbf{I}(o \# c1))$.

To ensure that all members of a subclass are also members of its superclasses, that is, $o \# c1$ and $c1 \## scl$ imply $o \# scl$, the following restriction is imposed:

– For all $o, c1, scl \in \mathbf{D}$, if $TVal_{\mathcal{I}}(o \# c1) = TVal_{\mathcal{I}}(c1 \## scl) = \mathbf{t}$ then $TVal_{\mathcal{I}}(o \# scl) = \mathbf{t}$.

5. *Frame*: $TVal_{\mathcal{I}}(o[a_1->v_1 \dots a_k->v_k]) = \mathbf{I}_{\text{truth}}(\mathbf{I}(o[a_1->v_1 \dots a_k->v_k]))$. Since the bag of attribute-value pairs represents the conjunctions of all the pairs, the following restriction is used, if $k > 0$:

– $TVal_{\mathcal{I}}(o[a_1->v_1 \dots a_k->v_k]) = \mathbf{t}$ if and only if
 $TVal_{\mathcal{I}}(o[a_1->v_1]) = \dots = TVal_{\mathcal{I}}(o[a_k->v_k]) = \mathbf{t}$.

6. *Externally defined atomic formula*: $TVal_{\mathcal{I}}(\text{External}(t)) = \mathbf{I}_{\text{truth}}(\mathbf{I}_{\text{external}}(t))$.

7. *Conjunction*: $TVal_{\mathcal{I}}(\text{And}(c_1 \dots c_n)) = \mathbf{t}$ if and only if $TVal_{\mathcal{I}}(c_1) = \dots = TVal_{\mathcal{I}}(c_n) = \mathbf{t}$. Otherwise, $TVal_{\mathcal{I}}(\text{And}(c_1 \dots c_n)) = \mathbf{f}$.

The empty conjunction is treated as a tautology, so $TVal_{\mathcal{I}}(\text{And}()) = \mathbf{t}$.

8. *Disjunction*: $TVal_{\mathcal{I}}(\text{Or}(c_1 \dots c_n)) = \mathbf{f}$ if and only if $TVal_{\mathcal{I}}(c_1) = \dots = TVal_{\mathcal{I}}(c_n) = \mathbf{f}$. Otherwise, $TVal_{\mathcal{I}}(\text{Or}(c_1 \dots c_n)) = \mathbf{t}$.

The empty disjunction is treated as a contradiction, so $TVal_{\mathcal{I}}(\text{Or}()) = \mathbf{f}$.

9. *Quantification*:

– $TVal_{\mathcal{I}}(\text{Exists } ?v_1 \dots ?v_n(\varphi)) = \mathbf{t}$ if and only if for *some* \mathcal{I}^* , described below, $TVal_{\mathcal{I}^*}(\varphi) = \mathbf{t}$.

– $TVal_{\mathcal{I}}(\text{Forall } ?v_1 \dots ?v_n(\varphi)) = \mathbf{t}$ if and only if for *every* \mathcal{I}^* , described below, $TVal_{\mathcal{I}^*}(\varphi) = \mathbf{t}$.

Here \mathcal{I}^* is a semantic structure of the form $\langle \mathbf{TV}, \mathbf{DTS}, \mathbf{D}, \mathbf{D}_{\text{ind}}, \mathbf{D}_{\text{func}}, \mathbf{I}_c, \mathbf{I}_v^*, \mathbf{I}_r, \mathbf{I}_{\text{frame}}, \mathbf{I}_{\text{sub}}, \mathbf{I}_{\text{isa}}, \mathbf{I}_{\rightarrow}, \mathbf{I}_{\text{external}}, \mathbf{I}_{\text{truth}} \rangle$, which is exactly like \mathcal{I} , except that the mapping \mathbf{I}_v^* is used instead of \mathbf{I}_v . \mathbf{I}_v^* is defined to coincide with \mathbf{I}_v on all variables except, possibly, on $?v_1, \dots, ?v_n$.

10. *Rule implication*:

– $TVal_{\mathcal{I}}(\text{conclusion} :- \text{condition}) = \mathbf{t}$, if either $TVal_{\mathcal{I}}(\text{conclusion}) = \mathbf{t}$ or $TVal_{\mathcal{I}}(\text{condition}) = \mathbf{f}$.

– $TVal_{\mathcal{I}}(\text{conclusion} :- \text{condition}) = \mathbf{f}$ otherwise.

11. *Groups of rules*:

If Γ is a group formula of the form $\text{Group}(\varphi_1 \dots \varphi_n)$ then

– $TVal_{\mathcal{I}}(\Gamma) = \mathbf{t}$ if and only if $TVal_{\mathcal{I}}(\varphi_1) = \mathbf{t}, \dots, TVal_{\mathcal{I}}(\varphi_n) = \mathbf{t}$

– $TVal_{\mathcal{I}}(\Gamma) = \mathbf{f}$ otherwise.

This means that groups of rules are treated as conjunctions. □

Interpretation of Documents. *RIF* documents are sets of rules and are similar to Group formulas in that respect. The main difference is that group formulas are used mainly as an anchor for attaching meta-information to sets of rules, while document formulas can also import other documents. This feature requires special care, since the imported documents can have `rif:local` constants and there might be name clashes. This is

a problem, if one recalls that `rif:local` constants are supposed to be completely encapsulated within the containing documents, so the same local constant may mean different things in different documents. Thus, RIF documents also define scope for local constants.

The locality property of `rif:local` constant symbols is achieved with the help of the notion of *semantic multi-structures*. Semantic multi-structures are collections of nearly identical regular semantic structures, but they can differ where it matters. This notion is defined next.

Definition 8 (Semantic multi-structure). A semantic multi-structure is a set $\{\mathcal{J}, \mathcal{K}, \mathcal{I}_1^\varphi, \dots, \mathcal{I}_n^\varphi, \dots\}$ of semantic structures, where \mathcal{J} and \mathcal{K} the regular semantic structures and the structures \mathcal{I}_i^φ are adorned with distinct RIF-BLD documents $\varphi_1, \dots, \varphi_n$ (adorned structures can be thought of as formula-structure pairs). All these structures are required to be identical in all respects except that the mappings $\mathbf{J}_C, \mathbf{K}_C, \mathbf{I}_1^C, \dots, \mathbf{I}_n^C, \dots$, which have the form $\text{Const} \rightarrow \mathcal{D}$, may differ on the constants that belong to the `rif:local` symbol space. \square

Everything is now ready for introducing the semantics of RIF documents.

Definition 9 (Truth valuation for document formulas). Let Δ be a document formula and let $\Delta_1, \dots, \Delta_k$ be all the RIF-BLD document formulas that are *imported* directly or indirectly (i.e., through some other imported document) into Δ . Let $\Gamma, \Gamma_1, \dots, \Gamma_k$ denote the respective group formulas associated with these documents. Let $\mathcal{I} = \{\mathcal{J}, \mathcal{K}, \mathcal{I}^{\Delta_1}, \dots, \mathcal{I}^{\Delta_k}, \dots\}$ be a semantic multi-structure that contains semantic structures adorned with at least the documents $\Delta_1, \dots, \Delta_k$. Truth valuation for RIF-BLD documents is now defined as follows:

$$TVal_{\mathcal{I}}(\Delta) = \mathbf{t} \text{ if and only if} \\ TVal_{\mathcal{K}}(\Gamma) = TVal_{\mathcal{I}}^{\Delta_1}(\Gamma_1) = \dots = TVal_{\mathcal{I}}^{\Delta_k}(\Gamma_k) = \mathbf{t}. \quad \square$$

Observe that this definition considers only those document formulas that are reachable via the one-argument import directives. Two-argument import directives are covered in [Sect. 10.5](#).

Note also that some of the Γ_i above may be missing since group subformulas in a document formula are optional. In this case, it is assumed that Γ_i is a tautology and every $TVal$ function maps such a Γ_i to the truth value \mathbf{t} .

For non-document formulas, $TVal_{\mathcal{I}}(\varphi)$ is extended from regular semantic structures to multi-structures as follows: if $\mathcal{I} = \{\mathcal{J}, \mathcal{K}, \dots\}$ is a multi-structure, then $TVal_{\mathcal{I}}(\varphi) = TVal_{\mathcal{J}}(\varphi)$.

Note the role of the different semantic structures in a multi-structure $\{\mathcal{J}, \mathcal{K}, \mathcal{I}^{\Delta_1}, \dots, \mathcal{I}^{\Delta_k}, \dots\}$: the structure \mathcal{K} is used to interpret the main document Δ . The structures \mathcal{I}^{Δ_i} are used for the documents that Δ imports, and the structure \mathcal{J} is used to interpret non-document formulas that Δ entails. \square

These definitions make the intent behind the `rif:local` constants clear: occurrences of such constants in different documents can be interpreted differently even

if the constants have the same name. Therefore, each document can choose the names for the `rif:local` constants freely and without a risk of name clashes with imported documents.

Logical Entailment. Finally, the stage is prepared for the notion of logical entailment. Here, the main interest lies in the entailment of RIF condition formulas, which play the role of queries to RIF-BLD documents.

Definition 10 (Models). A multi-structure \mathcal{I} is a *model* of a formula, φ , written as $\mathcal{I} \models \varphi$, iff $TVal_{\mathcal{I}}(\varphi) = \mathbf{t}$. Here φ can be a document or a non-document formula. \square

Definition 11 (Logical entailment). Let φ and ψ be (document or non-document) formulas. Then φ entails ψ , written as $\varphi \models \psi$, if and only if for every multi-structure \mathcal{I} , $\mathcal{I} \models \varphi$ implies $\mathcal{I} \models \psi$. \square

It is interesting to note that one consequence of the multi-document semantics of RIF-BLD is that local constants are completely encapsulated within their containing documents and cannot be queried from other documents. For instance, if one document, Δ' , has the fact

```
"http://example.com/p"^^rif:iri("abc"^^rif:local)
```

while another document formula, Δ , imports Δ' and has the rule

```
"http://example.com/q"^^rif:iri(?X) :-
    "http://example.com/p"^^rif:iri(?X)
```

then $\Delta \models$ `"http://example.com/q"^^rif:iri("abc"^^rif:local)` does *not* hold. This is because the symbol `"abc"^^rif:local` in Δ' and in Δ is treated as occurrences of different constants by semantic multi-structures.

The behavior of local symbols should be contrasted with that of `rif:iri` symbols, which are global. For instance, in the above scenario, let Δ' also have the fact `"http://example.com/p"^^rif:iri("http://cde"^^rif:iri)`. Then $\Delta \models$ `"http://example.com/q"^^rif:iri("http://cde"^^rif:iri)` *does* hold.

10.4.4 Implementation Challenges

Despite its conceptual simplicity (after all, it is just Horn clause logic), RIF-BLD is a challenging language to implement. There are four main difficulties:

- Equality
- Datatypes
- Interaction with RDF
- Interaction with OWL

Issues with equality. The difficulty with equality in RIF-BLD is that this predicate is allowed to appear in the rule heads. For instance,

```
Document (Prefix(ex <http://example.com>)
  Group (
    ex:foo = ex:bar
    ex:f(?X)=ex:g(?Y ex:h(?Z)) :- And(ex:p(?X ?Z) ex:q(?Z ?Y))
  )
```

According to the principle of substitution of equals by equals (also known as the congruence property of equality), any occurrence of `ex:foo` in a formula can be replaced with `ex:bar` without changing the meaning of the formula: whenever the original formula is true so must be the substituted version, and vice versa. Since function symbols are allowed, substitution can happen at any level of nesting, which means that, whenever an equality is derived, every formula has to be searched for occurrences of the terms at each side of the derived equality, and substitution must be performed. If this does not sound hard enough, consider the equalities that might be inferred by the second rule above. This rule entails equality among function terms that can have completely different structure. So, when searching for occurrences of the equated terms, complex matching must be performed. Things get even more involved when the terms to be matched contain variables.

To make the long story short, theorem proving with equality is known to be very hard, and there is an entire field, Term Rewriting [38], which investigates ways of dealing with a simplified version of equality, *directional* equality.

Despite all these difficulties, some cases of equality reasoning are easier to implement. One case is when the equality predicate occurs either in the rule premises or as ground (i.e., variable-free) facts that equate constants only. In that case, rules can be rewritten to incorporate equality as follows:

- Add an additional universal fact: `forall ?X (?X=?X)` and a rule that expresses the transitivity property of equality.
- If a variable or a constant, `v`, occurs in the rule, replace it with a new variable `?V` and add `?V=v` as a conjunct in the rule premises.

These rewritten rules can be made to work reasonably efficiently if supported by an adequate indexing scheme for the rules and facts. With a little more work, this method can be extended to allow function symbols in the equality predicates as well. For instance, a (potentially expensive) preprocessing step could apply the congruence axiom to the equality facts after which the above rewriting will work correctly even if function terms are equated.

Another well-known technique is to restrict equality to constants only, but to also allow the equality predicate to occur in any rule head, not just the facts. Function symbols are allowed everywhere, but the method is sound and complete only if the only derived equality happens to be among constants. This method is based on the union-find algorithm [39]. The union-find algorithm is used to maintain sets of constants that

have been inferred to be equal. The sets are maintained using tree structures that are equipped with a fast operation for finding the root of each tree. When members of different sets are inferred to be equal, the sets are merged. With appropriate data structures, the unification algorithm, which underlies most implementations of rule systems, can be modified so that each constant is replaced by the root of its respective tree.

Very few of the current rule systems support equality. The Vampire theorem prover [40] provides a complete implementation of equality. The FLORID system [41] provides partial implementation based on the union-find algorithm mentioned above. The FLORA-2 system [23] also has a partial implementation, which is based on a version of the rule rewriting technique mentioned above.

Issues with datatypes. Recall that a datatype in RIF is a symbol space whose constants are interpreted by a special subdomain of D_{ind} (see Definition 6). For instance, the members of the datatype of integers are constants of the form `"1"^^xs:integer`, `"2"^^xs:integer`, etc. The constant `"1"^^xs:integer` is interpreted by the mapping I_C as the actual integer 1, the integer constant `"2"^^xs:integer` is interpreted using the actual integer 2, etc. Now, what if there is a document of the following form?

```
Document (Prefix(ex <http://example.com>)
  ex:p("1"^^xs:integer "2"^^xs:integer)
  ?X=?Y :- ex:p(?X ?Y)
)
```

Logically, one should derive `"1"^^xs:integer = "2"^^xs:integer`. Since these constants are interpreted by the actual integers 1 and 2, respectively, the derived fact must be false in every semantic structure. This means that the above RIF document is unsatisfiable even though the document consists of Horn clauses alone! Checking for such contradictions is a very expensive operation in rule systems.

Integration with RDF. The main issue with RIF+RDF integration is that RDF has so-called blank nodes. As will be seen in [▶ Sect. 10.5.1](#), this makes it possible to derive existentially quantified facts on the RIF side. For querying, this is not a very serious problem, since existential variables can be skolemized, that is, replaced with new constants. As far as querying is concerned, skolemization is sound and complete. The problem arises not in RIF-BLD itself, but in its extensions. For example, if a RIF dialect allows aggregate functions, such as counting (see RIF-FLD [22]), then it is unclear what such a counting operation should mean. Suppose that both `p(a)` and `Exists ?X p(?X)` are derived. How many tuples does `p` have? A new theory of aggregate functions would be necessary to support such systems.

Integration with OWL. Integration with OWL is a far more difficult problem. First, this integration essentially includes the entire SWRL language [42], which is known to be undecidable and hard to implement. More importantly, since RIF-BLD allows function symbols, the combined RIF-BLD/OWL language is actually much larger than SWRL and no implementation of such a language has been reported so far.

10.4.5 Built-ins and Semantic Puzzles

One noteworthy implication of the decision to use general semantic structures for RIF-BLD and RIF-FLD was the emergence of certain semantic “puzzles” having to do with built-ins.

The first problem has to do with the generality of syntax. Although all built-ins used by RIF-BLD [28] are either functions or predicates, RIF anticipates that some dialects might define built-ins that use some other types of terms – for instance, frame terms or equality terms. This makes the semantic side of the built-ins (i.e., external terms) harder to define. The requirement for such generality has led to the concept of *external schemas* for built-ins. This chapter does not provide the details, but they can be found in [22, 28].

The second problem has to do with the fact that, unlike in logic programming and databases, defining the actual built-ins is not that simple. For instance, how should one define the predicate `is-literal-string`, which is true precisely of strings? In logic programming (and databases), the semantics is defined using Herbrand domains [35] where constant symbols stand for themselves. So, if one sees the domain element `1` then he can tell that it is an interpretation of an integer just by looking at the syntax. Similarly, if one looks at the domain element `f(abc)`, then one knows that it is not an integer, since it looks different. This simplicity does not extend to RIF for two reasons:

- Constants are normally *not* members of the domain of interpretation and they do not stand for themselves (so the above trick will not work).
- RIF has the equality predicate.

To see what equality has to do with built-ins, consider the question

Is "http://example.com/123"^^rif:iri an integer?

Surprisingly, the answer is, *it depends*. In some semantic structures, the above `rif:iri` constant might be mapped to, say, number 1; in some others it might be mapped to a person named John; and in yet others to something else. Moreover, if the set of formulas entails `"http://example.com/123"^^rif:iri = "1"^^xs:integer` then the answer is definitely “yes:” the above constant is an integer and is indistinguishable from 1.

Fortunately, the above problem is not serious: it was brought up just to illustrate a point. Since the domain of integers is a well-defined subset of the domain of semantic structures in RIF and since $\mathcal{I}(\text{is-literal-string}(c)) = I_{\text{External}}(\text{is-literal-string})(I_C(c))$, one simply needs to check if $\mathcal{I}(c)$ is a member of the domain of integers in order to determine the truth value of such a predicate.

The built-in predicate `iri-string` is a harder case to crack. Intuitively, `iri-string("http://foo.com"^^rif:iri "http://foo.com"^^xs:string)` should be true, but this better be false: `iri-string("http://bar.com"^^rif:iri "http://foo.com"^^xs:string)`. It turns out that one can ensure the former, but not the latter!

To appreciate the problem, recall that $I_C(\text{"http://foo.com"^^rif:iri})$ is just an element of the domain of interpretation D , which may have no syntactic attributes of the actual `rif:iri` constant `"http://foo.com"^^rif:iri` (since D can be any set). So, how should

$I_{\text{External}}(\text{iri-string})(I_C(\text{"http://foo.com"}^{\wedge\wedge\text{rif:iri}} \text{"http://foo.com"}))$

be interpreted? This can be postulated to be true because $I_C(\text{"http://foo.com"}^{\wedge\wedge\text{rif:iri}})$ is an image of the constant $\text{"http://foo.com"}^{\wedge\wedge\text{rif:iri}}$. However, one *cannot* force

$I_{\text{External}}(\text{iri-string})(I_C(\text{"http://bar.com"}^{\wedge\wedge\text{rif:iri}} \text{"http://foo.com"}))$

to be false because $I_C(\text{"http://foo.com"}^{\wedge\wedge\text{rif:iri}})$ might be the same element as $I_C(\text{"http://bar.com"}^{\wedge\wedge\text{rif:iri}})$ in D . In fact, if a document entails the formula

$\text{"http://foo.com"}^{\wedge\wedge\text{rif:iri}} = \text{"http://bar.com"}^{\wedge\wedge\text{rif:iri}}$

then in any model of this document the following combinations are all true:

```
iri-string("http://foo.com"^^rif:iri "http://foo.com"^^xs:string)
iri-string("http://foo.com"^^rif:iri "http://bar.com"^^xs:string)
iri-string("http://bar.com"^^rif:iri "http://bar.com"^^xs:string)
iri-string("http://bar.com"^^rif:iri "http://foo.com"^^xs:string)
```

The final subtlety to be mentioned here has to do with the notion of an *error*. In procedural languages and even in logic programming languages, built-in functions and predicates have intended domains, and passing values outside of those domains causes errors that may abort execution. However, an “abort” is not a suitable notion for a declarative language like RIF-BLD. To illustrate the problem, consider the formula

$3^{\wedge\wedge\text{xs:int}} = \text{numeric-add}(1^{\wedge\wedge\text{xs:int}} \text{"http://foo.com"}^{\wedge\wedge\text{rif:iri}})$

First, the earlier discussion suggests that one cannot tell a priori whether this should be true, false, or an error – it depends on what the arguments of `numeric-add` are mapped to by I_C . Second, suppose $I_C(\text{"http://foo.com"}^{\wedge\wedge\text{rif:iri}})$ is mapped to an element in the domain of the datatype `xs:string`. Clearly, this should be an error, but what is “error” in logic anyway?

There is no consensus on how errors should be modeled in logic languages. One possibility is to add a special truth value, but this is a very complex solution, and it is unclear how it might affect future dialects. The solution chosen by the RIF working group was simple and elegant: when an argument to a built-in predicate or function is mapped by I_C to an element in a wrong domain (like the domain of strings instead of numbers in the above example), the value of the corresponding built-in is *indeterminate*. That is, the RIF Datatypes and Built-ins Document [28] do not prescribe what the value of the corresponding built-in function (or the truth value of the predicate) should be: in some semantic structures it could be one thing and in others it could be something else. Therefore, while exchanging rules, RIF dialects must not make any assumptions about the values of built-ins when wrong arguments are passed to them. Put another way, RIF engines should not rely on any particular behavior in case of erroneous invocations of built-ins.

10.5 RDF and OWL Compatibility

The Rule Interchange Format (RIF) is a format for exchanging rules over the Web. These rules may reference external data sources that are based on different data models and represented using languages different from RIF. Of particular interest are the Resource Description Framework (RDF) [1], RDF Schema (RDFS) [43], and the OWL Web Ontology Language [44].

Normally RDF data, RDFS, and OWL ontologies are represented using RDF graphs, which are exchanged using the normative RDF/XML syntax [45]. RIF does not provide a format for exchanging RDF graphs. Instead, it assumes that these graphs are exchanged using RDF/XML. This chapter uses the commonly accepted triple syntax, *subject-property-object*, to represent edges in an RDF graph.

A typical scenario for the use of RIF with RDF/OWL is the exchange of rules that use RDF data and/or RDFS or OWL ontologies: an interchange agent *A* has a rules language that is RDF/OWL-aware, that is, it can use RDF data and RDFS/OWL ontologies directly. Agent *A* sends agent *B* some rules represented using the RIF/XML syntax, possibly with references to some RDF graph(s). *B* receives the rules and retrieves the referenced RDF graph(s). The rules are translated to the internal rules language of *B* and are processed together with the RDF graphs using the RDF/OWL-aware rule engine of *B*. Instead of being sent to a particular agent, *A* might publish the rules on a website. Multiple consumers can then download those rules.

Another important use of the RIF interface to RDF/OWL is that this defines standardized RIF-RDF and RIF-OWL rule languages, which might be preferable to nonstandard community-driven rule languages such as SWRL [42].

The technical device used to combine RIF rules and RDF/OWL graphs is the `Import` directive, which was discussed in [♦ Sect. 10.4.1](#). The RIF-RDF and OWL Compatibility document [26] specifies the semantics of such a combination. In particular, it specifies the logical consequences (i.e., the entailment relation) of such a combination. This is done by connecting the model theory of RIF (see [♦ Sect. 10.4](#)) with the model theories of RDF [46] and OWL [47], respectively.

The RDF semantics specification [46] defines four normative notions of entailment for RDF graphs: *simple*, *RDF*, *RDFS*, and *datatype*. OWL 2 also specifies two different semantics, with corresponding notions of entailment: the *direct* semantics [47], which specifies OWL 2 DL entailment, and the *RDF-based* semantics [48], which specifies OWL 2 Full entailment. The RIF-RDF and OWL Compatibility document defines how RIF interoperates with RDF/OWL for all six notions. [♦ Section 10.5.1](#) is concerned with the combination of RIF and RDF/RDFS. The combination of RIF and OWL is addressed in [♦ Sect. 10.5.2](#). The semantics of the interaction between RIF and OWL 2 DL is close in spirit to [42].

In RIF, the way to specify the particular semantics under which RDF and OWL documents are to be used is to import them with appropriate *profiles*. The profile is given as the second argument to the `Import` directive. Recall from [♦ Sect. 10.4.3](#) that RIF defines the semantics only for the one-argument `Import` directive.

The two-argument directive is reserved for importing documents that are “foreign” to RIF and the semantics of such directives are expected to be defined by separate specifications that concern themselves with combining RIF with those types of documents. The RIF RDF and OWL Compatibility specification thus addresses the semantics of two-argument `Import` directives that refer to RDF and OWL documents. For example, the statement

```
Import (<http://xmlns.com/foaf/spec/index.rdf>
       <http://www.w3.org/2007/rif-import-profile#RDFS>)
```

imports the FOAF ontology using the RDFS profile, that is, the statements in the ontology are interpreted using RDFS semantics.

10.5.1 Combination with RDF

The basic idea behind combining RDF graphs with RIF is that RDF subject–property–object triples of the form $s\ p\ o$ are mapped to RIF frame formulas of the form $a\ [p\ \rightarrow\ o]$. That is, the semantics is such that the RDF triple is satisfied if and only if so is the corresponding RIF frame formula. In addition, the triples of the form $s\ rdfs:type\ o$ and $s\ rdfs:subclassOf\ o$ correspond to the RIF membership and subclass formulas $s\ \#o$ and $s\ \#\#o$, respectively. For instance, if the RDF triples

```
ex:John foaf:Document ex:mobile
ex:Bill foaf:Document ex:mobile
ex:mobile rdf:type foaf:Project
```

are true then so are the RIF formulas

```
ex:John[foaf:Document -> ex:mobile]
ex:Bill[foaf:Document -> ex:mobile]
ex:mobile # foaf:Project
```

If, in addition, the RIF document has the rule

```
?pers[foaf:knows->?pers2] :-
    And(?pers[foaf:Document->?proj]
        ?pers2[foaf:Document->?proj]
        ?proj # foaf:Project)
```

then $ex:John[foaf:knows\ \rightarrow\ ex:Bill]$. $ex:Bill[foaf:knows\ \rightarrow\ ex:John]$ can also be derived, and the corresponding RDF triples are also true. Note that the above example uses the compact URI notation introduced in Definition 4 (see the *prefix* directive in item 6). Here *ex* is some suitably chosen compact URI prefix, such as <http://example.com>.

In reality, things are slightly more complex, since the syntax of RIF constant symbols slightly differs from that of RDF (this was necessary due to the generality of

RIF compared to RDF), but this difference can be disregarded for most practical purposes – especially since RIF also provides syntactic shortcuts that erase most of the differences. The exact correspondence between the RIF symbols and the symbols in RDF/OWL can be found in [26].

One subtlety concerning the above correspondence is that RDF has so-called blank nodes – a special syntax for existential variables which allows to denote such a variable using a single symbol without any explicit quantification. RIF does not have such special syntax and, instead, uses a more general syntax borrowed from first-order logic. As a result, RDF triples that have blank nodes cannot be mapped to RIF frame formulas directly. However, they correspond to existential RIF condition formulas. For example, given an RDF triple,

```
ex:John foaf:surname _:x
```

where `_:x` represents a blank node, the following RIF formula is entailed:

```
Exists ?X (ex:John[foaf:surname->?X])
```

A *RIF-RDF combination* is a pair $\langle R, S \rangle$ where R is a RIF document and S is a set of RDF triples. The central notion in the semantics of such a combination is the notion of a *common RIF-RDF interpretation*, which is a pair $\langle \hat{\mathcal{I}}, I \rangle$, where $\hat{\mathcal{I}}$ is a RIF semantic multi-structure for R , as in Definition 8, and I is a *simple* interpretation (for RDF graphs), as defined by the RDF Semantics specification [46]. The gory details are in how these semantic structures are stitched together. Rather than boring the reader with the details (which can be found in [26]), they can be explained at a high level. One just needs to keep in mind that what RIF calls constant symbols corresponds to RDF's IRIs and literals. In addition, recall the correspondence between RDF's triples and RIF's frame and membership/subclass formulas. Therefore, at a high level, the stitching between RIF semantic multi-structures and RDF interpretations can be described as follows:

- The (union of the) sets of resources and properties in the simple RDF interpretation I coincides with the set of individuals \mathcal{D}_{ind} in $\hat{\mathcal{I}}$.
- The extension of RDF properties is related to the mapping I_{frame} , so that triples get identified with frame formulas.
- The extent of the `rdf:type` property is related to the mapping I_{isa} in $\hat{\mathcal{I}}$, so that the triples of the form `s rdf:type o` would become equivalent to RIF membership formulas of the form `s#o`.
- Similarly, the extent of the `rdfs:subClassOf` property is related to the mapping and in this way the triples of the form `s rdfs:subClassOf o` become identified with RIF subclass formulas `s##o`.
- In addition, other elements of these languages (e.g., datatypes, lists) are related appropriately.

A pair $\langle \hat{\mathcal{I}}, I \rangle$ is said to be a *common RIF-RDFS interpretation* if it is a common RIF-RDF interpretation and, in addition, I is an RDFS interpretation.

Definition 12 (RDF and RDFS models). A common RIF-RDF interpretation $\langle \hat{\mathcal{I}}, I \rangle$ is a *simple RIF-RDF model* of a combination $\langle R, S \rangle$ if $\hat{\mathcal{I}}$ is a model of R and I is a model of S .

$\langle \hat{\mathcal{I}}, I \rangle$ is a *RIF-RDF model* if it is a simple RIF-RDF model and, in addition, I is an RDF interpretation, that is, a simple interpretation that satisfies all the RDF semantic conditions as defined in the RDF model theory specification [46].

$\langle \hat{\mathcal{I}}, I \rangle$ is a *RIF-RDFS model* if it is a RIF-RDF model and I is an RDFS interpretation, that is, an RDF-interpretation that satisfies all the RDFS axiomatic triples and semantic conditions required by the RDFS semantics [46]. \square

The above three notions of models for the combined RIF-RDF language now give rise to three notions of entailment: simple, RDF, and RDFS. Namely, a RIF-RDF combination $\langle R, S \rangle$ *simple-entails* (respectively, *RDF-entails*, or *RDFS-entails*) a RIF condition formula ϕ if and only if every simple RIF-RDF-model (respectively, RIF-RDF-model, or RIF-RDFS-model) of $\langle R, S \rangle$ also satisfies ϕ .

10.5.2 Combination with OWL

RIF combinations with OWL 2 are defined analogously to RIF-RDF and RIF-RDFS combinations. The combination of RIF with OWL 2 Full is straightforward, and is defined first.

OWL 2 Full has an alternative semantics defined in terms of RDF graphs [48]. This semantics takes RDFS interpretations of OWL 2 Full documents and further restricts them by the semantic conditions of [Sect. 10.5](#) of [48]. Such interpretations will be called *OWL 2 RDF-based interpretations*.

Definition 13 (OWL 2 Full Models). Let $C = \langle R, S \rangle$ be a RIF-OWL 2 Full combination, that is, R is a RIF-BLD document and S is an OWL 2 ontology. Let $\langle \hat{\mathcal{I}}, I \rangle$ be a common RIF-RDFS interpretation (Definition 12) such that

- I is an OWL 2 RDF-based interpretation
- $\langle \hat{\mathcal{I}}, I \rangle$ is a RIF-RDFS model of C .

In this case, $\langle \hat{\mathcal{I}}, I \rangle$ is said to be a *RIF-OWL-Full model* of C . \square

RIF-OWL 2 Full entailment is now defined using RIF-OWL-Full models, as usual.

RIF combinations with OWL DL are harder to define because not every RDF graph is also an OWL DL ontology. For example, unlike RIF and OWL Full, OWL DL does not view properties and classes as constants. Instead, they are conceptualized as binary and unary relations. Among other things, this implies that variables *cannot* range over classes and properties. This implies that not every RIF-BLD document can be combined with OWL DL: to be combinable, RIF documents must satisfy certain restrictions.

Recall that RDF triples are represented in RIF as frame formulas as well as isa- and subclass-formulas. If such a triple is part of an OWL DL ontology, the aforesaid properties of OWL DL imply certain restrictions on RIF formulas. A RIF atomic formula, ϕ , is called a *RIF-OWL-DL formula* if either of the following holds:

- ϕ is a frame formula of the form $a[b_1 \rightarrow c_1 \dots b_n \rightarrow c_n]$ such that $n > 1$, all the b_i s are constants, and
 - if any of the b_i is `rdf:type`, then c_i is a constant symbol.
 - if any of the b_i is `rdfs:subClassOf`, then a and c_i are constant symbols.
- ϕ is an isa-formula of the form `o # c` and c is a constant.
- ϕ is a sub-formula of the form `s##c` and both s and c are constants.

A *RIF-OWL-DL combination* is now defined as a pair $\langle R, O \rangle$ where R is a RIF document where every atomic formula is a RIF-OWL-DL formula and S is an OWL-DL ontology.

RIF-BLD semantic structures must also be restricted slightly to ensure that properties and classes are not interpreted as individuals. Details of these restrictions are not important and can be found in [26].

Next, *common RIF-OWL-DL interpretations* are defined analogously to the common RIF-RDF interpretations; that is, they are pairs $\langle \hat{I}, I \rangle$ such that \hat{I} is a RIF semantic structure (restricted as explained above) and I is the usual OWL DL interpretation [47]. In addition, \hat{I} and I are stitched together similar to the case of RDF. The notion of models is also defined by analogy with the previous cases.

Definition 14 (RIF-OWL-DL models). Let $C = \langle R, S \rangle$ be a RIF-OWL-DL combination and let $\langle \hat{I}, I \rangle$ be a *common RIF-OWL-DL interpretation* such that

- I is an OWL DL model of S
- \hat{I} is a RIF-BLD model of R . □

RIF-OWL-DL entailment is defined using RIF-OWL-DL models, as usual.

When using RIF together with OWL DL, the reader should be aware of certain subtleties that arise due to the assumptions underlying OWL DL. For instance, OWL DL postulates the disjointness of classes and datatypes, object properties and datatype-valued attributes, and so on. To illustrate this phenomenon, consider the OWL DL ontology

```
ex:John rdf:type ex:Person
```

and the following RIF fact:

```
ex:John # xs:string
```

These two statements do not have a common RIF-OWL-DL model because OWL assumes disjointness of datatypes, like `xs:string`, and classes, such as `ex:Person`. For another example, consider the OWL DL ontology

```
ex:spouse rdf:type owl:ObjectProperty
```

In OWL, an object property is a binary relation whose range are individuals, and individuals are disjoint from the elements of datatypes, such as `xs:string` and `xs:integer`. Consider now a RIF fact of the form

```
ex:Mary[ex:spouse -> "John"^^xs:string]
```

Here, the value of the property `ex:spouse` for the object `ex:Mary` is a string "John" and, as mentioned above, this stands in contradiction with the OWL DL statement that `ex:spouse` is an object property. Therefore, the above OWL DL and RIF statements do not have a common RIF-OWL-DL model and these statements are not satisfiable together.

10.6 An Example Application in RIF

Continuing with the story from [Sect. 10.2](#), recall the group of hackers locked in a garage who are using RIF in the quest to put together an aggregator site for mobile phone service procurement. The system consists of several parts:

- *Parts 1 and 2*: The knowledge bases of the service providers and manufacturers
- *Part 3*: Profiles of the customers
- *Part 4*: FOAF-like information maintained independently
- *Part 5*: The rules and queries defined by the phone service aggregator

Parts 1–5 are sources of information that are likely to have been created independently of each other and, for the purpose of this example, assume that they (at least Parts 1, 3, and 5) are represented using RIF. The knowledge bases of the manufacturers and service providers might or might not be proprietary, but one can assume that the service aggregator has access to that information (possibly for a fee). The rules and queries defined by the service aggregator are likely to be proprietary, while the FOAF-like information is public. To provide better advice, the service aggregator takes its customers through a series of questions, which leads to creation of a profile for each customer. These profiles must stay private, so Part 3 really consists of many different private sources, one per customer. Since the customers' profiles are rather simple in nature, these profiles can be represented in RDF.

Consider now Parts 1 and 2 of the system. Note that most of the IRIs used in the examples are simply made up and do not represent real entities. In some cases, Wikipedia references are used, as the required concepts are yet to be "webified." To improve the visual appeal, the example takes advantage of the syntactic shortcuts allowed by the RIF Datatypes and Built-ins specification [28]. Thus, for instance, "abcd" is used instead of "abcd" `"xs:string` and 123 instead of "123" `"xs:integer`. Compact URIs, introduced earlier, are also used extensively.

```
Document (Prefix (wp <http://en.wikipedia.org/wiki/>)
                Prefix (ph <http://example.com/Phones/>))
Group (
  wp:MobilePhone ## wp:Phone
  wp:Smartphone ## wp:MobilePhone

  ph:BBCurve8830 # wp:Smartphone
  ph:iPhone # wp:Smartphone
  ph:PalmPre # wp:Smartphone
  ph:PalmCentro # wp:Smartphone
```



```

ph:BBCurve8320 # wp:MobilePhone
ph:BBCurve8330 # wp:MobilePhone
ph:MotoRAZR # wp:MobilePhone

ph:BBCurve8320 [ph:network-> wp:GSM
                ph:feature-> wp:WiFi
                ph:feature-> wp:Camera
                ph:kbdtype-> "hard"
                ph:kbdtype-> "full" ]
ph:BBCurve8330 [ph:network-> wp:CDMA
                ph:feature-> wp:Camera
                ph:kbdtype-> "hard"
                ph:kbdtype-> "full" ]
ph:BBCurve8830 [ph:network-> wp:CDMA
                ph:network-> wp:GSM
                ph:feature-> wp:WiFi
                ph:feature-> wp:Camera
                ph:feature-> wp:Gps
                ph:kbdtype-> "hard" ]
ph:iPhone [ph:network-> wp:GSM
           ph:feature-> wp:WiFi
           ph:feature-> wp:Camera
           ph:feature-> wp:Gps
           ph:feature-> wp:Touchscreen]
ph:PalmPre [ph:network-> wp:CDMA
            ph:feature-> wp:WiFi
            ph:feature-> wp:Camera
            ph:feature-> wp:Gps
            ph:feature-> wp:Touchscreen
            ph:kbdtype-> "hard" ]
ph:PalmCentro [ph:network-> wp:CDMA
               ph:network-> wp:GSM
               ph:feature-> wp:Camera
               ph:feature-> wp:Touchscreen
               ph:kbdtype-> "hard" ]
ph:MotoRAZR [ph:network-> wp:CDMA
             ph:network-> wp:GSM
             ph:feature-> wp:Camera
             ph:kbdtype-> "hard" ]

... ..

))

```

The phone knowledge base might also include various rules. They are shown separately, outside of the above document, in order to avoid layout problems. One rule could be that smartphones support browsing and e-mail, and they offer full keyboards. This is why these features are not mentioned explicitly.

```
?X[ph:feature-> wp:Email
  ph:feature-> wp:Www
  ph:kbdtype-> "full" ] :- ?X # wp:Smartphone
```

It can also be safely assumed that if a smartphone does not offer a hardware keyboard then it offers a software keyboard:

```
?X[ph:kbdtype-> "soft" ] :-
  And(?X # wp:Smartphone Naf ?X[ph:kbdtype-> "hard" ])
```

The above rule is a telling example where RIF-BLD falls short: *Naf*, which stands for *negation as failure* [34], is not part of that dialect. Many deductive Semantic Web systems support this feature [23, 24, 49, 50], so *Naf* is envisioned to be part of a future dialect that extends RIF-BLD, and this operator is supported by the RIF Framework for Logic Dialects [22]. It might seem plausible that the use of *Naf* could have been prevented if one uses classical negation and, for example, OWL to represent the phone information. However, this would have required that one lists all the missing features explicitly for every phone. For instance, the phone manufacturers would have to say explicitly that BBCurve8330 and MotoRAZR are not smartphones, that they do not have GPS, that the Centro has no Wi-Fi, that the iPhone has no hardware keyboard, and so on. Apart from the inconvenience of having to explicitly mention and maintain heaps of negative information, it is well-known that classical negation cannot replace all the uses of *Naf* [51].

Next, consider Part 2, the service provider knowledge base. Clearly, the picture presented here is greatly simplified in order to make the example manageable within the confines of this chapter. Also, as before, the service plans and the cost information are completely fictional.

```
Document (Prefix (wp <http://en.wikipedia.org/wiki/>)
  Prefix (mo <http://example.com/MobileOperators/>)
  Prefix (ph <http://example.com/Phones/>)
  Import (<http://example.com/Phones>))
Group (
  mo:GoPhone[mo:service-> mo:prepaid
    ph:network-> wp:GSM]
  mo:ATT[mo:service-> mo:contract
    mo:network-> wp:GSM]
  mo:Tmobile[mo:service-> mo:contract
    mo:service-> mo:prepaid
    ph:network-> wp:GSM]
```

```

mo:Verizon[mo:service-> mo:contract
    mo:service-> mo:prepaid
    ph:network-> wp:CDMA]
mo:Sprint[mo:service-> mo:contract
    ph:network-> wp:CDMA]
mo:Boost[mo:service-> mo:prepaid
    ph:network-> wp:CDMA]

mo:plan(mo:GoPhone mo:prepaid 1)[mo:phone-> ph:MotoRAZR
    mo:cost-> 20
    mo:minutes-> 100]
mo:plan(mo:ATT mo:contract 1)[mo:phone-> ph:PalmCentro
    mo:phone-> ph:iPhone
    mo:phone-> ph:BBCurve8830
    mo:cost-> 70
    mo:minutes-> 500
    mo:data-> "true" xs:boolean]
mo:plan(mo:Tmobile mo:contract 1)[mo:phone-> ph:BBCurve8320
    mo:phone-> ph:MotoRAZR
    mo:phone-> ph:BBCurve8830
    mo:cost-> 50
    mo:minutes-> 400]
mo:plan(mo:Tmobile mo:prepaid 1)[mo:phone-> ph:BBCurve8320
    mo:phone-> ph:MotoRAZR
    mo:cost-> 20
    mo:minutes-> 100]
mo:plan(mo:Sprint mo:contract 1)[mo:phone-> ph:BBCurve8330
    mo:phone-> ph:PalmCentro
    mo:cost-> 50
    mo:minutes-> 450]
mo:plan(mo:Sprint mo:contract 2)[mo:phone-> ph:PalmCentro
    mo:phone-> ph:PalmPre
    mo:phone-> ph:BBCurve8330
    mo:cost-> 60
    mo:minutes-> 450
    mo:data-> "true"^^xs:boolean]
mo:plan(mo:Boost mo:prepaid 1)[mo:phone-> ph:BBCurve8330
    mo:phone-> ph:PalmCentro
    mo:cost-> 45
    mo:minutes-> 400]
mo:plan(mo:Boost mo:prepaid 2)[mo:phone-> ph:PalmCentro
    mo:cost-> 55
    mo:minutes-> 400
    mo:data-> "true"^^xs:boolean]

```

```

mo:plan(mo:Verizon mo:contract 1) [mo:phone-> ph:BBCurve8330
                                mo:cost-> 50
                                mo:minutes-> 400]
mo:plan(mo:Verizon mo:contract 2) [mo:phone-> ph:BBCurve8330
                                mo:phone-> ph:BBCurve8830
                                mo:phone-> ph:PalmCentro
                                mo:cost-> 60
                                mo:minutes-> 400
                                mo:data-> "true"^^xs:boolean]
mo:plan(mo:Verizon mo:prepaid 1) [mo:phone-> ph:BBCurve8330
                                mo:cost-> 20
                                mo:minutes-> 100]
... ..
))

```

Note that the service plans document imports the document identified by the IRI <http://example.com/Phones>, and it is assumed that this IRI represents the location of the phones document described earlier (recall that the first argument to `Import` must be the location of the imported document).

What kind of rules might be part of the service provider knowledge base? One could be as follows. Suppose the user made a selection of a plan and a phone. The phone might be compatible with the plan, but the combination is not ideal. For instance, the selected phone might be a smartphone, but the plan does not offer unlimited data (e.g., Palm Centro and the first plan from Sprint). Or, vice versa, the plan offers unlimited data, but the phone cannot take advantage of it (e.g., Blackberry 8330 and the second plan from Sprint). The following rule takes a plan and a phone (which come from the aggregator) and offers alternative plans from the same provider (and of the same type: contract or prepaid), if the phone and the plan do not match:

```

?plan[mo:alternative(?phone)-> ?altplan] :-
  And(?plan = mo:plan(?company ?plantype ?planId)
      ?altplan = mo:plan(?company ?plantype ?altPlanId)
      ?altplan[mo:phone-> ?phone]
      Or(And(Naf ?plan[mo:data-> "true"^^xs:boolean]
              ?phone # ph:Smartphone
              ?altplan[mo:data-> "true"^^xs:boolean])
         And(Naf ?phone # ph:Smartphone
              ?plan[mo:data-> "true"^^xs:boolean]
              Naf ?altplan[mo:data-> "true"^^xs:boolean]))

```

Another rule could be used to suggest an alternative phone if a selected plan and phone do not match well. Likewise, given a selected phone, a rule could suggest matching plans or suitable phones could be offered for a selected plan. Here is an example of a rule that could be used to offer matching plans:

```
?phone[mo:plan -> ?plan] :-
  And(?phone # wp:MobilePhone
    ?plan[mo:phone-> ?phone]
    Or(And(?phone # wp:Smartphone
      ?plan[mo:data-> "true"^^xs:boolean])
      And(Naf ?phone # wp:Smartphone
        Naf ?plan[mo:data-> "true"^^xs:boolean])))
```

The rule defines a new property for mobile phones, `mo:plan`, which ranges over the plans that are compatible with the phone and offers just the right service (i.e., offer data service for smartphones and do not offer it for regular phones).

Part 3 deals with descriptions of the profiles that are created for the customers of the aggregator site. Apart from the name, Id, and similar properties, assume that the customers tell the aggregator the number of close friends that they talk to and the cell phone companies that those friends use. They also specify the number of minutes that they use talking to their friends and the number of minutes they use talking to others. Based on this information and the information contained in Parts 1 and 2, the aggregator can try to suggest an optimal plan based on the customer requirements and costs. Here is a sample customer profile. Because RDF supports only binary relations (triples), the profile representation is a bit awkward (it uses mobile company names as properties) and might earn a frown from a serious database designer.

```
<!-- Prefixes: cu <http://example.com/Aggregator/Customers/> -->
<!-- Prefixes: ag <http://example.com/Aggregator/> -->

cu:John mo:Sprint 5
cu:John mo:ATT 7
cu:John mo:Verizon 3
cu:John ag:friendMinutes 300
cu:John ag:otherMinutes 200

cu:Mary mo:Verizon 6
cu:Mary mo:Sprint 7
cu:Mary mo:Boost 4
cu:Mary mo:GoPhone 7
cu:Mary ag:friendMinutes 400
cu:Mary ag:otherMinutes 400
```

With this information, the aggregator can try to determine the actual number of payable minutes for each plan and then compute the effective cost of each plan for any given customer. Again, the intent here is to illustrate the point rather than attempting to mimic the actual ways that phone companies use to compute chargeable minutes. For the purpose of this example, assume that the only non-chargeable minutes are the ones that are made between phones in the same network (a common perk in the USA). As before, RIF-BLD is inadequate, if one needs to express the requisite rules, so features

borrowed from RIF-FLD save the day again. In this instance, one needs aggregate functions. Recall that RDF triples of the form $s p o$ correspond to RIF's frames of the form $s[p-\>o]$, and this is how customers' profiles are referred to in the example. Also note that profiles are imported using the two-argument `Import` directive, which specifies the simple-entailment semantics for the imported RDF document. The following RIF document represents a fragment of what could be Part 5 of the procurement application:

```
Document (Prefix (mo <http://example.com/MobileOperators/>)
  Prefix (ph <http://example.com/Phones/>)
  Prefix (ag <http://example.com/Phones/>)
  Prefix (ag <http://example.com/Aggregator/>)
  Prefix (f <http://www.w3.org/2007/rif-builtin-function#>)
  Import (<http://example.com/MobileOperators>)
  Import (<http://example.com/Aggregator/Customers/Profiles>
    <http://www.w3.org/2007/rif-import-profile#Simple>))

Group (
  "minutes"^^rif:local (?customer ?plan ?mins) :-
    And (?customer [ag:friendMinutes->?FM ag:otherMinutes->?OM]
      ?plan = mo:plan (?company ?type ?id)
      ?allFriends =
        sum {?F[?customer] | Exists ?co (?customer[?co->?F])}
      ?nonNetFriends =
        sum {?F[?customer ?company] |
          Exists ?co (And (?customer[?co->?F]
            Naf ?co = ?company))}
      ?mins = External (
        f:numeric-add (
          ?OM
          External (
            f:numeric-multiply (
              ?FM
              External (
                f:numeric-divide (?nonNetFriends
                  ?allFriends))))))

  ?plan [ag:cost (?customer) -> ?cost] :-
    And ("minutes"^^rif:local (?customer ?plan ?chargemins)
      ?plan [mo:cost-> ?basecost mo:minutes-> ?baseminutes]
      ?cost = External (
        f:numeric-max (
          ?basecost
          External (
```

```

      f:numeric-multiply(
        ?basecost
        External(
          f:numeric-divide(?chargemins
                           ?baseminutes))))))
    ... ..
  ))

```

The first rule defines the amount of chargeable minutes for a given customer and a selected plan. The predicate in the rule head is not expected to be used outside of the given document, so one can use the local constant "minutes"^^rif:local as its name. The rule uses two aggregate functions to compute the total number of friends and the number of friends that would be on a different network, if the customer chooses the current plan. RIF built-in functions [28] are then used to compute the actual number of chargeable minutes by reducing the number of minutes allocated to calling friends by the proportion of out-of-network friends. The variables that occur inside the brackets [...] in the aggregate functions (e.g., [?customer]) are the *grouping variables* and are similar to the grouping variables in, say, SQL. Their precise meaning in RIF can be found in [22]. The variable ?F inside the aggregates is the aggregation variable. It is scoped within the corresponding aggregate term, so the two occurrences of ?F actually refer to different variables.

The keyword `External` above signifies that the term is a built-in function and thus is going to be interpreted accordingly. The reader should not be fazed by the number of nested `Externals` in the example. At some point the working group had plans for introducing various shortcuts that would make the presentation syntax more appealing, but these intentions did not come to fruition. One should not lose sight of the fact that RIF is intended for rule exchange and the presentation syntax always had limited purposes in RIF.

The second rule above is the one that computes the actual cost of the given plan for the customer. It does not introduce new features as far as RIF is concerned. It simply uses the first rule to determine the number of chargeable minutes and then (greatly simplifying the picture) defines the cost of a plan as a proportion of the plan's base cost. The term

```
mo:plan(mo:ATT mo:contract 1) [ag:cost (cu:John) -> 70]
```

is an example of what this rule will compute.

Part 4 of the aggregator's system is not formally developed in this chapter, as such an exercise does not present interesting opportunities for the illustration of new important features of RIF. Instead, it might be worthwhile to speculate about how the procurement application could be enhanced by taking into account public information, such as FOAF [10]. One possibility is that the aggregator might try to help customers to determine who they are calling frequently. For instance, using the FOAF data, the aggregator could use a rule that would suggest people known to the customer that have the same hobbies, who are involved in the same projects, and so on. This idea can be taken further if one lets imagination run wild and assumes that, in a not-so-distant future, community websites,

such as Facebook and LinkedIn, might start offering enhanced FOAF-style semantic information (possibly restricted to preserve privacy or for a fee). In fact Facebook's Open Graph is a step toward this and this is covered in [Semantic Annotation and Retrieval: Web of Hypertext – RDFa and Microformats](#).

In conclusion of this example, it is worth mentioning another important feature that is provided by RIF-FLD, but is not part of the Basic Logic Dialect: *remote modules*. Recall that the procurement application imports customer profiles all at once. What if customers were able to specify more complex profiles? For instance, a well-designed interface could help each customer to create *rules*, not just RDF facts. In that case, loading all profiles at once would be problematic, since rules made by one customer could possibly derive information based on the data associated with a different customer. This could be prevented by turning each profile into a separate document and importing just the right profile for each customer.

Still, there may be a problem if one takes this scenario a step further. The aggregator might decide to develop an application that finds optimal plans for groups of people, where the cost is minimized for groups rather than for each individual separately. In that case, the RIF document for Part 5 might need to import several profiles at once, and the problem of unforeseen interaction among rules re-emerges. The notion of remote modules in RIF-FLD is designed to prevent this kind of problems by *encapsulating* rules within their respective documents [22]. The following modified fragment of the earlier document for Part 5 illustrates the use of remote modules.

```
Document (
  Prefix (mo <http://example.com/MobileOperators/>)
  Prefix (ph <http://example.com/Phones/>)
  Prefix (ag <http://example.com/Aggregator/>)
  Prefix (f <http://www.w3.org/2007/rif-builtin-function#>)
  Import (<http://example.com/MobileOperators>)
  Module (_J <http://example.com/Aggregator/Customers/Profiles/
    John>)
  Group (
    "minutes"^^rif:local(?customer ?plan ?mins) :-
      And(?customer[ag:friendMinutes->?FM ag:otherMinutes->?OM]@_J
        ?plan = mo:plan(?company ?type ?id)
        ... .. .
      )
  )
)
```

Instead of importing John's profile, it is linked here as a remote module through the `Module` directive. The second argument to the directive represents the location of the module, while the first is a RIF constant (here a `rif:local` constant `_J`) used to refer to the module from within the document. The statement `?customer[ag:friendMinutes->?FM ag:otherMinutes->?OM]@_J` is a query to John's profile viewed as a remote knowledge base. The rules and the facts of that knowledge base are not physically imported

into the document and are not directly accessible to the rules of that document. Any number of such remote modules can be connected to a RIF document without the danger of inferential interference, that is, the possibility that facts contained in or derived by one knowledge base might enable further (possibly unintended) derivations using the rules of another knowledge base. Such encapsulation is not provided by the import mechanism.

RIF-FLD defines only a mechanism for statically attached remote modules. However, dialects defined using RIF-FLD are free to introduce executable rules whose purpose would be to declare new modules dynamically, as part of the inference. It is this latter mechanism that would be most useful for the above example, since customer identities are likely to be read from the input and thus their profiles would have to be linked dynamically.

10.7 Conclusions

This chapter gave an overview of the recently adopted Rule Interchange Format standard – the latest installment in a series of Semantic Web standards – and illustrated how RIF can be used for building sophisticated applications on the Semantic Web, which can reason over distributed collections of data and rules. Most of the specifications produced by the RIF Working group [4] became W3C Recommendations in June 2010.

At the time of this writing, the curtain is about to fall on the activities of the RIF working group, but the work on the implementation of the *vision* of RIF is far from over. At present, only three dialects are official W3C recommendations: RIF-BLD (Horn logic), RIF-PRD (production rules), and their common subset RIF-Core. However, most of the typical applications of rules in the past 30 years require facilities that go far beyond Horn logic. The phone service procurement application, which was used to motivate and illustrate RIF in this chapter, relies on features that are well outside of the capabilities of RIF-BLD.

Various user communities have plans to develop more powerful RIF dialects based on RIF-FLD, the RIF Framework for Logic Dialects. One important actor in this field is the RuleML initiative (<http://ruleml.org>), which already published several additional RIF dialects – see http://www.w3.org/2005/rules/wiki/RIF_FLD_Dialects. Two very important extensions of RIF-BLD developed by the RuleML community introduce the `NaF` operator, which was used in the procurement example. One extension uses the *well-founded* semantics [13] for `NaF` and is described in <http://ruleml.org/rif/RIF-CLPWD.html>. The other is based on the *stable model* semantics [14]; it can be found in <http://ruleml.org/rif/RIF-CASPD.html>. More powerful extensions are also being contemplated. For instance, aggregate operators, like the ones used in [Sect. 10.6](#), are extremely important and are prime targets for future dialects. Rules that can deal with uncertain, fuzzy, and probabilistic data are also very important. One such dialect has already been developed as part of RuleML; it can be found in http://ruleml.org/rif/URSW2008_F9_ZhaoBoley.pdf. Another important extension that is being planned is related to defeasible reasoning and argumentation theories [52], as in the SILK rule language [24].

Acknowledgments

The author wishes to thank Jos de Bruijn for the discussions that helped shape the contents of this chapter. The author is also grateful to the members of the RIF Working Group, especially Harold Boley, for their time and contributions. Much of the material in this chapter is based on the work of that group: RIF Basic Logic Dialect [20], RIF-RDF and OWL Compatibility [26], and RIF-FLD [22]. Further thanks goes to Harold Boley, Paul Fodor, and Hui Wan who have read an early version of this chapter and gave useful suggestions. Last, but not least, thanks goes to the editors who devoted much effort to putting together this handbook and applying finishing touches to its chapters, including this one.

References

1. Klyne, G., Carroll, J.J.: Resource Description Framework (RDF): concepts and abstract syntax, W3C Recommendation (Feb 2004)
2. Dean, M., Schreiber, G.: OWL web ontology language reference, W3C Recommendation (Feb 2004)
3. Duerst, M., Suignard, M.: Internationalized Resource Identifiers (IRIs). <http://www.ietf.org/rfc/rfc3987.txt> (2005)
4. W3C: RIF: Rule Interchange Format working group. http://www.w3.org/2005/rules/wiki/RIF_Working_Group (2010)
5. Enderton, H.: A Mathematical Introduction to Logic. Academic, Boston (2001)
6. Mendelson, E.: Introduction to Mathematical Logic. Chapman & Hall/CRC Press, Boca Raton (1997)
7. Clocksin, W., Mellish, C.: Programming in Prolog. Springer, New York (1981)
8. Kifer, M., Lausen, G., Wu, J.: Logical foundations of object-oriented and frame-based languages. *J. ACM* **42**, 741–843 (1995)
9. Chen, W., Kifer, M., Warren, D.: HiLog: a foundation for higher-order logic programming. *J. Log Program.* **15**(3), 187–230 (1993)
10. FOAF: The friend of a friend (FOAF) project. <http://www.foaf-project.org/>
11. Boag, S., Chamberlin, D., Fernandez, M., Florescu, D., Robie, J., Simeon, J., Ste-fanescu, M.: Xquery 1.0: an xml query language, W3C Technical Report. <http://www.w3.org/XML/Query> (2004)
12. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF, W3C Proposed Recommendation (Nov 2007)
13. Van Gelder, A., Ross, K., Schlipf, J.: The well-founded semantics for general logic programs. *J. ACM* **38**(3), 620–650 (1991)
14. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: Proceedings of the Fifth Conference and Symposium on Logic Programming (ICLP 1988), Seattle, pp. 1070–1080. MIT Press, Cambridge (1988)
15. Jess: Jess, the rule language for the Java platform. <http://herzberg.ca.sandia.gov/jess/> (2008)
16. JBoss: Drools. <http://labs.jboss.com/drools/> (2010)
17. WebSphere ILOG JRules. <http://www-01.ibm.com/software/integration/businesses-rule-management/jrules/> (2010)
18. Bry, F., Eckert, M., Patranjan, P.L.: Reactivity on the web: paradigms and applications of the language XChange. *J. Web Eng.* **5**(1), 3–24 (2006)
19. Kozlenkov, A., Penaloza, R., Nigam, V., Royer, L., Dawelbait, G., Schroeder, M.: PROVA: a language for rule-based Java scripting, data and computation integration, and agent programming. In: Proceedings of the 10th International Conference on Extending Database Technology (EDBT 2006), Munich. Lecture Notes in Computer Science, vol. 4254, pp. 899–908. Springer, Berlin (2006)
20. Boley, H., Kifer, M.: RIF basic logic dialect, W3C Recommendation. <http://www.w3.org/TR/rif-bl/> (June 2010)

21. de Sainte Marie, C., Paschke, A., Hallmark, G.: RIF production rule dialect. <http://www.w3.org/2005/rules/wiki/PRD/> (2009)
22. Boley, H., Kifer, M.: RIF framework for logic dialects, W3C Recommendation. <http://www.w3.org/TR/rif-fld/> (June 2010)
23. Kifer, M.: FLORA-2: an object-oriented knowledge base language. The FLORA-2 web site. <http://flora.sourceforge.net>
24. Vulcan, Inc.: The SILK project: semantic inferencing on large knowledge. <http://silk.semwebcentral.org/> (2010)
25. Kifer, M.: Rule interchange format: the framework. In: Calvanese, D., Lausen, G. (eds.) *Web Reasoning and Rule Systems*, Second International Conference (RR 2008), Karlsruhe. *Lecture Notes in Computer Science*, vol. 5341, pp. 1–11. Springer, Heidelberg (2008)
26. de Bruijn, J.: RIF RDF and OWL compatibility, W3C Recommendation. <http://www.w3.org/TR/rif-rdf-owl/> (June 2010)
27. Boley, H., Hallmark, G., Paschke, M.K.A., Polleres, A., Reynolds, D.: RIF core dialect, W3C Recommendation. <http://www.w3.org/TR/rif-core/> (June 2010)
28. Polleres, A., Boley, H., Kifer, M.: RIF datatypes and built-ins 1.0, W3C Recommendation. <http://www.w3.org/TR/rif-dtb/> (June 2010)
29. Paschke, A., Hirtle, D., Ginsberg, A., Patranjan, P.L., McCabe, F.: RIF use cases and requirements, W3C Working Draft. <http://www.w3.org/TR/rif-ucr/> (Dec 2008)
30. Mitchell, S., Morgenstern, L., Paschke, A.: RIF test cases, W3C Working Draft. <http://www.w3.org/TR/rif-test/> (June 2010)
31. Kifer, M.: Rules and ontologies in F-Logic. In: Eisinger, N., Maluszynski, J. (eds.) *Reasoning Web*, First International Summer School, Msida, 25–29 July 2005. *Lecture Notes in Computer Science*, vol. 3564, pp. 22–34. Springer, Heidelberg (2005)
32. Bray, T., Paoli, J., Sperberg-McQueen, C.M., Maler, E., Yergeau, F., Cowan, J.: *Extensible markup language (XML) 1.1*, 2nd edn, W3C Recommendation (Aug 2006)
33. Fallside, D., Walmsley, P.: *XML schema part 0: primer* second edition. Technical report, WWW Consortium. <http://www.w3.org/TR/xmlschema-0/> (Oct 2004)
34. Clark, K.: Negation as failure. In: Gallaire, H., Minker, J. (eds.) *Logic and Data Bases*, pp. 292–322. Plenum Press, New York (1978)
35. Lloyd, J.: *Foundations of Logic Programming*, 2nd edn. Springer, New York (1987)
36. Biron, P.V., Malhotra, A.: *XML schema part 2: datatypes* second edition, W3C Recommendation (Oct 2004)
37. Yang, G., Kifer, M., Zhao, C.: FLORA-2: A rule-based knowledge representation and inference infrastructure for the semantic web. In: *Proceedings of the Second International Conference on Ontologies, Databases and Applications of Semantics (ODBASE 2003)*, Catania. *Lecture Notes in Computer Science*, vol. 2888, pp. 671–688. Springer, Heidelberg (2003)
38. Plaisted, D.: Equational reasoning and term rewriting systems. In: Gabbay, D., Hogger, C., Robinson, J. (eds.) *Handbook of Logic in Artificial Intelligence and Logic Programming*, vol. 1, pp. 273–364. Oxford University Press, Oxford (1993)
39. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: *Introduction to Algorithms*. MIT Press, Cambridge (2001)
40. Voronkov, A.: *Theorem proving and Vampire*. The Vampire web site. <http://www.voronkov.com/vampire.cgi> (2010)
41. Frohn, J., Himmeröder, R., Lausen, G., May, W., Schleppehorst, C.: Managing semistructured data with FLORID: a deductive object-oriented perspective. *Inform. Syst.* **23**(8), 589–613 (1998)
42. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosz, B., Dean, M.: SWRL: A semantic web rule language combining OWL and RuleML, W3C Member Submission (May 2004)
43. Brickley, D., Guha, R.V.: *RDF vocabulary description language 1.0: RDF schema*, W3C Recommendation (Feb 2004)
44. Motik, B., Patel-Schneider, P.F., Parsia, B.: *OWL 2 web ontology language structural specification and functional-style syntax*, W3C Working Draft (Apr 2009)
45. Beckett, D.: *RDF/XML syntax specification (revised)*, W3C Recommendation (Feb 2004)
46. Hayes, P.: *RDF semantics*, W3C Recommendation (Feb 2004)
47. Motik, B., Patel-Schneider, P.F., Cuenca-Grau, B.: *OWL 2 web ontology language direct semantics*, W3C Working Draft (Apr 2009)

48. Schneider, M.: OWL 2 web ontology language rdf-based semantics, W3C Working Draft (Apr 2009)
49. Ontoprise, GmbH: Ontobroker. <http://www.ontoprise.com/>
50. Sagonas, K., Swift, T., Warren, D.S., Freire, J., Rao, P., Cui, B., Johnson, E., de Castro, L., Marques, R.F., Dawson, S., Kifer, M.: The XSB system version 3.2: programmer's manual (2009)
51. Motik, B., Horrocks, I., Rosati, R., Sattler, U.: Can OWL and logic programming live together happily ever after? In: Proceedings of the Fifth International Semantic Web Conference (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 501–514. Springer, Heidelberg (2006)
52. Wan, H., Grosz, B., Kifer, M., Fodor, P., Liang, S.: Logic programming with defaults and argumentation theories. In: Proceedings of the 25th International Conference on Logic Programming (ICLP 2009), Pasadena (2009)



11 KR and Reasoning on the Semantic Web: Web-Scale Reasoning

Spyros Kotoulas¹ · Frank van Harmelen¹ · Jesse Weaver²

¹VU University Amsterdam, Amsterdam, The Netherlands

²Rensselaer Polytechnic Institute, Troy, USA

11.1	<i>Scientific and Technical Overview</i>	442
11.1.1	Harnessing Increased Computational Power	443
11.1.1.1	Parallel Strategies from AI	444
11.1.1.2	Distributed Semantic Web Reasoning	447
11.1.1.3	Peer-to-Peer Computing	448
11.1.1.4	Non-DHT-Based Peer-to-Peer Reasoning Systems	451
11.1.1.5	Federated RDF Stores	452
11.1.1.6	Cluster Computing	453
11.1.2	Containing the Number of Triples	454
11.1.2.1	Selection Strategies	454
11.1.2.2	Limiting the Scope of Statements	455
11.2	<i>Example Applications</i>	456
11.3	<i>Related Resources</i>	457
11.3.1	MaRVIN	457
11.3.2	WebPIE	458
11.3.3	Scalable Reduction	458
11.3.4	Cray XMT	459
11.3.5	LarKC	460
11.3.6	SAOR	461
11.4	<i>Future Issues</i>	461
11.5	<i>Cross-References</i>	463

Abstract: Reasoning is a key element of the Semantic Web. For the Semantic Web to scale, it is required that reasoning also scales. This chapter focuses on two approaches to achieve this: The first deals with increasing the computational power available for a given task by harnessing distributed resources. These distributed resources refer to peer-to-peer networks, federated data stores, or cluster-based computing. The second deals with containing the set of axioms that need to be considered for a given task. This can be achieved by using intelligent selection strategies and limiting the scope of statements. The former is exemplified by methods substituting expensive web-scale reasoning with the cheaper application of heuristics while the latter by methods to control the quality of the provided axioms. Finally, future issues concerning information centralization and logics vs information retrieval-based methods, metrics, and benchmarking are considered.

11.1 Scientific and Technical Overview

This chapter begins with two premises, each of which is barely disputed:

Premise 1: Inferences Are Key to the Semantic Web. A key element of the Semantic Web architecture is that not only are data published on the Web, but that these data are accompanied with schema information in the form of ontologies. The point of these ontologies is that they capture the intended semantics of the data in a form that allows inferences to be made that are not explicitly stated in the data themselves. But what does it mean to “capture the intended semantics”?

Although the primary definition of the semantics of formal languages is most often in terms of a denotational semantics (see, e.g., [17] for RDF and [36] and [KR and Reasoning on the Semantic Web: OWL](#)), perhaps a more productive definition on the Semantic Web is to describe semantic interoperability in terms of shared inferences. It could be said that the amount of shared semantics (or semantic interoperability) between two agents is measured by the number of new facts that both agents subscribe to after having exchanged a given sentence. Due to a number of sanctioned inferences, the agents will agree on more than just the exchanged sentence: They will agree on a number of inferences drawn from the exchanged sentence. And the larger and richer their shared inferences, the more semantically interoperable the agents are.

Premise 2: Inferences on the Semantic Web Must Scale. The logics on which the Semantic Web is based originate from work on databases and knowledge bases. However, the scale at which these inferences must be drawn and the number of premises that must be taken into account are many orders of magnitude larger than anything ever seen in either databases or knowledge bases.

These two premises have given rise to a large amount of research, technology, and development in the past few years on how to enable inferencing at very large scale, and ultimately at “web-scale.” This leads us to the central question:

- ▶ “How can Semantic Web reasoning scale?”

In the literature of the past decade, the following main directions can be identified that have been used to approach this central question:

- *Increase computational power* Be that in the form of computing grids, at-home computing, or by exploiting parallel and distributed architectures, the performance of reasoning methods can be improved by using more powerful hardware or additional computational nodes.
- *Contain the number of triples* that are involved in answering a query or inferring facts. Reducing scope limits the amount of data that needs to be considered to answer a query or perform inference. This can, for example, be accomplished by ontology modularization, triple ranking, or using information provenance.
- *Reduce query expressiveness* Another way to achieve scalability is to limit query expressiveness. Limiting query expressiveness allows using well-established methods from information retrieval and databases. This is common in Semantic Web Search Engines, which have been described in [▶ Semantic Web Search Engines](#).
- *Improve indexing and querying techniques* Breakthroughs in indexing technologies and organization of data can significantly improve performance. Similarly, better query planning and evaluation can reduce query latency and increase query throughput. These issues have already been extensively dealt with in [▶ Storing the Semantic Web: Repositories](#).
- *Use lighter logics* As described in [▶ KR and Reasoning on the Semantic Web: OWL](#), there are various language profiles for Semantic Web languages. As the expressive power of a profile decreases, computation becomes cheaper. These issues have been described in [▶ Semantic Annotation and Retrieval: RDF](#) and [▶ KR and Reasoning on the Semantic Web: OWL](#).

The last three of these options are being extensively discussed in other chapters of this book, as indicated. Hence, in this chapter, the focus will be on the first two approaches for enabling reasoning at web-scale.

11.1.1 Harnessing Increased Computational Power

A prominent way to scale reasoning to web-scale is by providing additional computational power. This can be achieved in two ways:

- *Faster hardware* The computational power of computers is constantly increasing. Thus, it is reasonable to expect that with existing methods it will be possible to process increasingly large datasets. This is inadequate for the following reasons:
 - The amount of data in the Semantic Web is increasing at a larger pace than the computational power of computers. Moore's law implies that computational power doubles every 2 years, linked open data doubles every 6 months. Refer to [▶ Semantic Annotation and Retrieval: Web of Data](#).
 - In recent years, processors have no longer become faster in terms of clock frequencies. The main drive to increase computational power is by using additional

computational cores. Practically all modern platforms now use a Symmetric Multi-processing (SMP) architecture. This means that reasoning methods developed for single processors will not immediately become faster with new hardware (<http://www.scidacreview.org/0904/html/multicore.html>).

Developments in hardware are not discussed any further, since they are out of the scope of this book.

- *Distribution* Parallel methods that can exploit distributed hardware can scale in one additional dimension: the number of computational nodes. The benefit in performance is orthogonal to the previous point, that is, as computers get faster, parallel approaches benefit equally with nonparallel approaches. The rest of this section will be devoted to improving performance through parallelization and distribution.

11.1.1.1 Parallel Strategies from AI

Needless to say, efficient reasoning has been a research problem since before the invention of the Semantic Web. As mentioned, the Semantic Web adds new challenges, of which a significant one is the vast amount of data published on the Web, or “web-scale data.” Exploiting distributed architectures is necessary for handling web-scale data, but parallelism must be employed to obtain decent performance on such architectures. This section provides a brief survey of forms of parallelism in reasoning systems. Although a detailed survey of (non-Semantic-Web) reasoning systems is beyond the scope of this chapter, the interested reader may refer to [49–51] for such surveys.

Bonacina [50] gives a classification of types of parallelism for deduction based primarily on the level of granularity, which can be broken down into three categories: parallelism at the term-, clause-, and search-levels. Of greatest interest to us is search-level parallelism, which accounts for aspects of distribution. In the following, we describe these broad categories and give some specific examples of deduction as well as Semantic-Web-related examples.

Term-Level Parallelism

Term-level parallelism is fine-grained and seeks to parallelize frequent, low-level operations. Examples of such operations include parallel term rewriting and parallel unification. Using parallel unification from Prolog systems as an example (explained in greater detail in [51]), consider two complex terms $p(a_1, \dots, a_m)$ and $q(b_1, \dots, b_n)$. If $p = q$ and $m = n$, then the two terms are potentially unifiable. To do so, for $1 \leq i \leq m = n$, attempt to $\text{unify}(a_i, b_i)$. Each $\text{unify}(a_i, b_i)$ can be executed in parallel with the caveat that dependencies must be handled within terms. For example, suppose there exists $j = i$ such that $a_i = a_j$. Then $\text{unify}(a_i, b_i)$ and $\text{unify}(a_j, b_j)$ may attempt to give different values to a_i and a_j (which are actually the same term). This is usually remedied by giving mutually dependent unification tasks to a single thread so that no coordination is required between threads.

Clearly, this level of parallelism is so fine-grained that performance gains are limited. Sometimes these types of operations are even implemented in hardware. For the Semantic

Web, terms are usually very simple with small arity. For example, interpreting the OWL 2 RL rules as Prolog rules, there is a single predicate T with arity three (representing a triple), and so individual unification operations can become at most three times faster when parallelized.

Clause-Level Parallelism

Clause-level parallelism is mid-grained (relative to the other two categories) and seeks to parallelize individual inference steps. The typical example is OR-parallelism in Prolog systems (discussed in great detail in [51]). Using SLD-resolution, given a query (or resolvent), a subgoal is selected, and clauses (Prolog rules) whose heads can unify with the subgoal are sought after. It is possible that there can be multiple such clauses, and each clause may possibly lead to a solution. Therefore, the subgoal can resolve with each clause in parallel, resulting in multiple resolutions of the original query. An example of OR-parallelism can be seen in the backward-chained, RDFS reasoning algorithm of Kaoudi, Miliaraki, and Koubarakis [25], to be discussed later in the chapter.

Search-Level Parallelism

Search-level parallelism is coarse-grained and seeks to divide the search space among processes. A key distinction of search-level parallelism is that each process has its own set of data. In term- and clause-level parallelism, it is implicit that the processes operate on shared data (for term-level parallelism, processes operate on parts of a shared clause; for clause-level parallelism, processes operate on a shared set of data). Thus, search-level parallelism lends itself well for distributed architectures. Another key distinction is that processes communicate with each other, whereas in term- and clause-level parallelism, communication is unnecessary since processes operate on shared data.

Up until now, the concept of a search strategy has been avoided, mainly because in the previous cases, parallelism was simple enough that such a formalism was unnecessary. However, at this point, it is necessary to formalize some concepts to help in making further distinctions. Loosely using the formalism of Bonacina [50], a search strategy C is a pair $\langle I, \Sigma \rangle$ where I is an inference system and Σ is a search plan. An inference system provides the rules for deduction, and a search plan provides a concrete method of applying the rules. In other words, a search plan decides how to deal with nondeterminism in the inferencing process. It determines at each inference step which rules should be applied and to what they should be applied. The search plan also determines when inferencing is finished and may impact decidability. For example, I could be an analytic tableau system and Σ would choose which rewrite rules to apply and to which axioms they should be applied at each inference step. (Note that Σ is not necessarily optimal in the sense that it always makes the best choice. Σ merely employs some method – often just some heuristics – to make a decision at each nondeterministic choice point.)

Search-level parallelism can be broken down into two categories. The first is whether homogeneous or heterogeneous inference systems are used. The second is the type of search employed, which may be multi-search or distributed-search (or both).

Homogeneous vs Heterogeneous In homogeneous inference systems, each process uses the same inference system I . This is the more common of the two approaches, and the systems discussed later in this chapter all fall under this category. Far more interesting, though, are heterogeneous inference systems in which each process uses a different inference system.

Rudolph, Tserendorj, and Hitzler [52] have proposed such a system for anytime, approximate reasoning. In a limited evaluation, they employed three algorithms for SHIQ DL reasoning: Screech-all, Screech-none, and KAON2. These systems are complete but unsound, sound but incomplete, and sound and complete, respectively. (The inference systems differ in how to handle rules with disjunctions in their heads.) Being modified to provide intermediate results at any given time, each algorithm is evaluated performing an instance retrieval task for varying lengths of time to determine the precision of their results over time. The precision of the three systems is then compared, and for that particular evaluation, Screech-none provided the best precision for the first 5 s, after which Screech-all provided better precision, until 6 s when KAON2 approaches sound and complete results. They then propose that for that specific problem, these algorithms could be run in parallel and if answers are requested before 5 s (the anytime behavior), then the intermediate results from Screech-none should be returned, and if after 6 s, the results of KAON2 should be returned. However, making such a determination (i.e., choosing the superior intermediate results) at any given time is difficult to determine, and so more extensive experimentation would need to be performed in order to develop better and more general heuristics.

Although no communication is used in the above example (i.e., processes do not collaborate), it is allowable for processes to do so in heterogeneous inference systems, perhaps to provide hints to each other.

Multi-Search In multi-search, each process employs a different search plan. That is, each process handles nondeterminism differently. As with heterogeneous inference systems, processes may communicate to share data or provide hints, but each process essentially attempts to solve the problem on its own. Assuming that each process has a fair search plan (i.e., the search strategy is decidable), then each process would correctly finish without needing to communicate with other processes. This should not be confused with heterogeneous inference systems, which derive parallelism by using multiple inference systems, whereas multi-search derives its parallelism from employing different approaches to handling nondeterminism in the inference system(s).

At this point, it is instructive to consider two types of nondeterminism: “don’t-know” and “don’t-care” nondeterminism. In don’t-know nondeterminism, the different paths of execution at a choice point may or may not lead to a solution, and so we “don’t know” which path to choose. If a path is chosen that does not lead to a solution, then backtracking occurs to the choice point so that another path can be chosen. This is different from don’t-care parallelism where each path leads to a solution, but the implications on performance may be different.

Therefore, in multi-search, if the search plans differ in how don't-know nondeterminism is handled, then the search itself is being parallelized. However, if the search plans differ in how don't-care nondeterminism is handled, then the search itself is not being parallelized but rather optimized. This is due to the fact that any path in don't-care nondeterminism leads to a solution, but one may lead to a solution more quickly than another. Thus, parallelism from don't-know nondeterminism can be viewed as a treasure hunt in which some competitors may not ever find the answer, but parallelism from don't-care nondeterminism can be viewed as a race in which all competitors get to the end, but only one gets there first. (In Answer Set Programming, parallelism derived from don't-know and don't-care nondeterminism has been referred to as vertical and horizontal parallelism, respectively [49].)

To be clear, in multi-search, processes may make the same decisions at some choice points; that is, the search plans may “overlap” in a sense. This differs from the previous example of OR-parallelism in which each process takes a single, unique path from a shared choice point.

Distributed-Search Distributed-search seeks to derive parallelism by subdividing the search space at every inference step. This subdivision is logically viewed as deciding which processes are “allowed” to produce what inferences. In actuality, it is a matter of decomposing the problem into parallel processes as in the parallel computing arena using communication between processes to exchange inferences/clauses/data in order to ensure completeness.

All of the forward-chained reasoning systems presented in the remainder of this chapter are examples of distributed-search.

11.1.1.2 Distributed Semantic Web Reasoning

Distribution refers to exploiting multiple computers to solve a problem collaboratively. It has a long tradition in high-performance computing, including applications such as chess playing, searching for extraterrestrial life, high-energy physics research, bioinformatics, and many more.

The following areas in distributed computing are relevant for scalable Semantic Web reasoning:

- *Peer-to-peer computing* Peer-to-peer technologies have a reputation of providing scalable infrastructures for large-scale data management problems. Their initial application domain was file sharing. Since then, a series of new applications have been proposed including anonymizing networks, voice-over-ip infrastructures, peer-to-peer television, and peer-to-peer Web servers. Some implementations and deployments have been very successful, for example, Skype, the tor anonymizer, and BitTorrent.
- *Federated stores* The idea of federating data sources predates the Semantic Web and was initially suggested in the field of databases [18]. As identified in [38], the main characteristics of a federated database are autonomy, heterogeneity, and distribution.

- *Cluster computing* Cluster-based methods were initially developed for high-performance computing. Modern cluster-based methods exploit architectures where tens to thousands of processors are interconnected through a fast network and are used in parallel to solve computationally difficult problem. Several architectures have been proposed [47]. Currently, the most popular architectures use a Hierarchical Memory System, where processors or processing cores share some memory segment while access to the rest is done through the network.
- *Distributed databases* Distributing and partitioning data among several tightly coupled stores have been a long-standing problem in the field of databases. In this book, the issue is covered in [Storing the Semantic Web: Repositories](#), so it will not be discussed here.

In the following subsections, the application of the aforementioned distributed computing techniques in the Semantic Web will be outlined.

11.1.1.3 Peer-to-Peer Computing

The current state of the art in peer-to-peer-based (P2P-based) Semantic Web infrastructures will be covered, focusing on systems and methods that scale or have the potential to scale to a large number of volumes of data or high throughput. The vast majority of peer-to-peer systems for Semantic Web reasoning are based in Distributed Hash Tables (DHTs) [28]. Triples are distributed among peers using a method that will be referred to as SPO-hashing. Query resolution has been investigated both for forward-chaining and backward-chaining reasoning.

DHTs DHTs are a well-researched type of a structured P2P system. Nodes function autonomously and collectively form a complete and efficient system without any central coordination. DHT overlays provide mappings from keys to objects, similar to a hashtable. Keys are chosen from a large space. This space is partitioned in zones, and each peer is responsible for the keys and corresponding objects in a zone. Peers need to maintain connections only to a limited number of other peers, and the overlay has the ability to self-organize, with respect to peer connections and object distribution, to handle network churn. In principle, all DHT-based systems provide the following functionalities: store (key, object), which stores an object identified by its key, and search (key), which returns the object (when it exists) from the peer responsible for the key. Current systems need approximately $O(\log(N))$ messages to search or store a mapping, and each peer needs to keep from $O(1)$ to $O(\log(N))$ pointers to other peers, where N is the number of peers in the network.

SPO-hashing First appearing in RDFpeers [8], DHTs can be used to maintain large triple indexes by terms or combinations of terms. Typically, triples are indexed by subject, predicate, object, amounting to a total of three indexes. Considering that DHTs can be used in a similar fashion with standard hashtables, maintaining such indexes is straightforward and scalable: Indexing a triple consists of inserting three entries into the system using S, P, or O as the DHT key and the remainder of the triple as the value.

To improve join and triple pattern lookup performance, this scheme has been extended as through auxiliary indexes. Work on [15] maintains additional indexes to increase the performance of lookup for triple patterns when more than one term is bound, inspired by [29]. In RDFcube [31], a 3D space is used with each dimension corresponding to the subject, the predicate, or the object of a triple used as a key. This key is mapped to an existence bit, which is in turn used to narrow down the candidate triples for joins.

Reasoning on DHT overlays Reasoning in DHT-based systems can either be done in a forward or backward manner. Forward-chaining or data-driven reasoning is triggered by new triples being inserted in the system while backward-chaining or goal-driven reasoning is triggered by queries. Clearly, there is a space-time trade-off between the two approaches. Forward chaining dictates that all inferred triples are precalculated and indexed at insertion time, requiring more space but simplifying querying. Backward chaining dictates that inference is driven by queries, requiring less space at the expense of additional cost at query time.

The description of current techniques for DHT-based reasoning will be streamlined by first describing forward chaining, then query resolution, and finally query resolution extended with backward reasoning.

First proposed by Heine et al. [19, 20], forward-chaining reasoning methods on top of DHTs have been limited to RDFS. All RDFS reasoning rules (see also ref to [KR and Reasoning on the Semantic Web: OWL](#)) are fired on triples that share at least one term, that is, all antecedents for any given rule share at least one term ([Table 11.1](#)). This means that, combined with SPO-hashing, the antecedents for any given rule all reside in the

Table 11.1

RDFS rules [17]. The terms that are shared between antecedents are underlined; namespaces have been omitted

1: $s p o$ (if o is a literal)	\Rightarrow $..n$ type Literal
2: \underline{p} domain x & $s \underline{p} o$	$\Rightarrow s$ type x
3: \underline{p} range x & $s \underline{p} o$	$\Rightarrow o$ type x
4a: $s p o$	$\Rightarrow s$ type Resource
4b: $s p o$	$\Rightarrow o$ type Resource
5: \underline{p} <u>subPropertyOf</u> q & \underline{q} <u>subPropertyOf</u> r	$\Rightarrow p$ subPropertyOf r
6: \underline{p} type Property	$\Rightarrow p$ subPropertyOf p
7: $s \underline{p} o$ & \underline{p} <u>subPropertyOf</u> q	$\Rightarrow s q o$
8: s type Class	$\Rightarrow s$ subClassOf Resource
9: s type \underline{x} & \underline{x} <u>subClassOf</u> y	$\Rightarrow s$ type y
10: s type Class	$\Rightarrow s$ subClassOf s
11: \underline{x} <u>subClassOf</u> y & \underline{y} <u>subClassOf</u> z	$\Rightarrow x$ subClassOf z
12: \underline{p} type ContainerMembershipProperty	$\Rightarrow p$ subPropertyOf member
13: o type Datatype	$\Rightarrow o$ subClassOf Literal

same peer, that is, the peer that stores all triples with the term in common. When a new triple is inserted in the system, either as input or as a result of reasoning, it is indexed using the SPO-hashing scheme; the reasoning rules are local. The triples that are inferred are sent to other nodes using the SPO-hashing scheme. This process is repeated until fixpoint. The generic algorithm for DHT-based forward reasoning is shown in Algorithm 1.

Algorithm 1 DHT-Based Forward Reasoning

```
//Each peer maintains a knowledge base  $K_p$ 
procedure INSERT(triple)
  send(triple, triple.subject)
  send(triple, triple.predicate)
  send(triple, triple.object)
end procedure

procedure RECEIVE(triple, peer)
  InferredTriples:=makeInference(triple,  $K_{peer}$ );
   $K_{peer} = K_{peer} \cup \text{triple}$ ;
  for all newTriple  $\in$  InferredTriples do
    send(newTriple, newTriple.subject)
    send(newTriple, newTriple.predicate)
    send(newTriple, newTriple.object)
  end for
end procedure

procedure SEND(triple, key)
  //implemented by the underlying DHT implementation
  targetPeer:=hash(key);
  receive(triple, targetPeer);
end procedure
```

Query resolution Using an SPO-hashing scheme on top of a DHT, queries with atomic triple patterns are easy to evaluate. If indexes for combinations of terms exist, it is enough to perform an index lookup for the bound terms in the pattern. In the basic scheme where only indexes on subject, predicate, and objects are maintained, a separate lookup should be issued for each bound term and the results should be joined.

Disjunctive queries can be implemented by splitting the query into subqueries and issuing them separately. The end result can be obtained by taking the union of the results of the subqueries.

Conjunctive queries are more challenging, since their evaluation requires joins over distributed data. To this end, a series of methods to do joins over DHTs have been proposed [6, 24, 27].

Backward chaining The work in [25] presents a backward-chaining algorithm for RDFS reasoning over DHTs based on predicate adornment. Furthermore, a comparison

with forward chaining is performed. The evaluation was made using a class hierarchy of up to 10,000 instances and indicates that storage cost for forward reasoning is at least one order of magnitude higher than for backward reasoning. On the other hand, querying cost is three to six times higher with backward reasoning.

Current issues The uptake of DHT-based reasoning techniques is inhibited by a series of problems, namely, data load balancing problems, inefficiency of distributed joins, and the lack of evaluation of current techniques.

Load balancing The distribution of term frequency in Semantic Web triples is very skewed [6, 26, 35, 45]. As mentioned in the SPO-hashing scheme and the description of DHTs, the DHT key space is divided into zones. Each zone is maintained by one peer and contains all the mappings (triples) for the keys (terms) that are contained in it. This means that a single peer will be called to store all triples that contain some (over-) popular terms. For example, there will be a single peer in the network, which will be called to store all triples with the term “rdf:type.” Additionally, it will be called to answer all queries for a triple pattern that contains “rdf:type.” Obviously, this will lead to severe load balancing problems.

There have been several proposals for load balancing on DHTs, mainly by distributing the mappings for overly popular keys among several nodes [7]. Nevertheless, these are not directly applicable to DHT-based Semantic Web reasoning, since this distribution will come at the expense of even more inefficient joins. Furthermore, it will break the premise that antecedents that share a term will be co-located at some peer. This will effectively mean that DHT-based forward reasoning will not be complete.

Work in [4] proposes a replication scheme where a collection of peers will be responsible for maintaining overly popular keys. They ensure completeness under RDFS semantics by replicating schema triples to all peers in the group. Unfortunately, this approach has not been evaluated.

Efficient joins and query planning Similar to querying traditional database systems, a key challenge with querying RDF stores lies in performing joins efficiently. When data are distributed, large amounts of data may need to be transferred over the network, hampering both throughput and response time. Another key issue is query planning. In a distributed setting, query planning becomes more challenging, since statistical information used for selectivity estimation heuristics is not centrally available [5].

Lack of evaluation Evidenced by the large number of relevant publications, DHT-based reasoning may be a promising direction for scalable DHT stores. Nevertheless, DHT-based reasoning systems have been very poorly evaluated, producing neither positive nor negative evidence about the practicality of these methods. Invariably, evaluation has been performed with very small datasets or by focusing on specific aspects of the problem. To date, there has been no large-scale deployment of DHT-based Semantic Web reasoning.

11.1.1.4 Non-DHT-Based Peer-to-Peer Reasoning Systems

A series of peer-to-peer reasoning systems do not use DHTs.

The work in [40] is based on the notion of path queries to build an index only on paths and subpaths, but not on individual elements for a data source. Every RDF model is seen

as a graph, where nodes correspond to resources, and arcs to properties linking these resources. The result of a query to such a model is a set of subgraphs corresponding to a path expression. Since the information that makes up a path might be distributed across different data sources, the index structure to use should also contain information about subpaths without losing the advantage of indexing complete paths, and the most suitable way to represent this index structure is a hierarchy, where the source index of the indexed path is the root element. In terms of space, the complexity of the index is $O(s * n^2)$, where s is the number of sources and n is the length of the schema path. The trade-off is that query answering without index support at the instance level is far more computationally intensive, so different techniques (partly similar to the ones used in [27], in terms of query chain formation and subquery ordering) are applied on the basis of an initial naive query-processing algorithm in order to perform join ordering and overall optimization, under the assumption that nodes do not have local join capabilities.

Bibster [13] follows an unstructured semantic-based P2P architecture: Each peer knows about its expertise and finds out about the expertise of neighboring peers through active advertising – thus, peers for expertise clusters. When a peer receives a query, it tries to answer it, or forward it to other peers whom it judges likely to be able to answer the query, based on similarity functions between the subject of the query and the previously advertised expertise topics using the schema hierarchy and text-similarity methods.

11.1.1.5 Federated RDF Stores

Federated RDF repositories aim at offering unified access among different RDF repositories by integrating them according to the federated repositories approach. Semantic Federations are collections of heterogeneous distributed RDF repositories that can be accessed as a unique local Semantic Repository.

Federated RDF stores consist of a network of autonomous repositories, which share interlinked data. The location of the data is largely determined by its publisher or owner. Typically, nodes maintain their own data and share metadata. The topology of the network is largely determined by the ontologies stored in each repository. To join the network, nodes are required to provide a mapping of their ontology to an ontology of a node already in the network, which then becomes their neighbor. Queries are posted on a local schema and routed to the nodes that can understand them (i.e., nodes with ontologies for which a mapping exists). Relevant work is presented in [1, 30, 37].

The advantages of federated RDF stores are that (a) data remains at its original location, which is beneficial for access control, security, and privacy and (b) they can integrate non-RDF data sources by providing adapters to RDF. Their disadvantages are that queries may need to be flooded, which is inefficient. For example, consider a query about some specific person using the FOAF schema. While most nodes in the network will be able to understand the query, very few will actually have an answer. Disseminating information about the instances of each node will lead to worse privacy characteristics.

Another disadvantage lies in that nodes are required to provide mappings to a node already in the network, which is sometimes very limiting.

A hybrid DHT-Federated datastore approach is presented in [2]. Peers keep control of their data, maintaining their own triples while a DHT-based index of peer contents is maintained. This index is consulted to locate peers with content relevant to some given query, which is in turn resolved in a manner similar to that of federated data stores.

Similar to DHT-based approaches, there has been neither extensive evaluation nor large-scale deployment of federated RDF stores.

11.1.1.6 Cluster Computing

Clusters consist of tens to hundreds of computational nodes communicating through a high-speed interconnect. Unlike peer-to-peer systems, nodes have similar computation capacity, good connectivity, and high availability. This makes the implementation of scalable, high-performance approaches more practical. Currently, large-scale distributed reasoning has only been deployed on such systems.

Soma and Prasanna [39] present a technique for parallel OWL inferencing through partitioning, inspired by similar techniques for Datalog [12]. They experiment with both data partitioning (each node gets a different partition of data and applies all rules) and rule partitioning (each node gets all data but applies only some rules), using different partitioning techniques. Experimental results show good speedup (both sub- and super-linear, depending on the dataset) but on relatively small datasets (1 M triples); runtime is not reported, nor scalability over larger datasets.

In [46], straightforward parallel RDFS reasoning on a supercomputer is presented. This approach replicates all schema triples to all processing nodes and distributes instance triples randomly. Each node calculates the closure of its partition using a conventional reasoner, and the results are merged. To ensure that there are no dependencies between partitions, triples extending the RDFS schema are ignored. The best reported result is the materialization of the closure of 345 M triples from the LUBM benchmark dataset is 8 min and 25 s on 128 processes.

The work on MaRVIN [26, 35] is motivated by the observation that it is hard to solve Semantic Web problems through traditional divide-and-conquer strategies since Semantic Web data are hard to partition. MaRVIN brings forward a method named divide-conquer-swap to do inferencing through forward chaining (i.e., calculate the closure of the input data): The input is divided into several partitions. These partitions are loaded by a number of peers, and the closure of the data contained in these partitions is calculated. The closure and the original data are re-partitioned and the process repeated until no new triples are derived. Experiments on real-world datasets show that MaRVIN can calculate the closure of 200 M triples on 64 compute nodes in 7.2 min, yielding a throughput of 450 K triples/s [26].

WebPIE (Web-scale Parallel Inference Engine) [44, 45] is a high-performance OWL reasoner using the MapReduce [10] parallel programming model. It performs RDFS and

OWL Horst reasoning on a cluster. It deals with issues of load balancing, data distribution, and unnecessary computation through a series of optimizations described in [Sect. 11.3](#). At the time of writing, WebPIE is the only system that has demonstrated Semantic Web reasoning over 100 billion triples.

Although the above approaches outperform state-of-the-art RDF stores in terms of loading and reasoning speed, it is important to note that they do not perform query answering. Thus, a direct comparison is not meaningful. With the exception of work in parallel databases, there exist no high-performance, highly distributed RDF stores with both querying and inferencing capabilities.

11.1.2 Containing the Number of Triples

The Semantic Web is inherently distributed and includes an open set of information providers. Thus, a model where all Semantic Web data are aggregated to a single point and reasoned over is unrealistic. In order to scale, an intelligent selection of the data to be reasoned over needs to be performed.

This can be achieved in several ways:

Firstly, subsets of triples can be selected before they are subjected to reasoning. Typically, in knowledge-based systems, the information flow is data-driven. This means that all information is first processed and indexed and then queries are processed. On a web-scale, this is impractical, since all information is very large and an open set. Instead, a model where search is interleaved with reasoning can be used, that is, additional information is fetched as reasoning progresses.

Secondly, information can be split in meaningful modules. Depending on the query, only the relevant modules will be loaded. Reasoning will be performed on this smaller dataset.

Thirdly, the scope of the statements made by an information source can be limited. In an open Web, irrelevant or erroneous statements can limit the scalability of the system. This can be avoided by limiting the scope of statements to their original context.

11.1.2.1 Selection Strategies

Although the challenge is often phrased as “reasoning at web-scale” (as is the title of this chapter), it is not at all always required to actually perform the reasoning at “web-scale.” Intelligent heuristics can often be used to limit the set of axioms (or facts, or triples) that need to be involved in the reasoning process. Such heuristics often judge at low costs the a priori relevance of a very large number of triples to a given task (e.g., answering a specific query), and then only feed those triples to a reasoner that survive such an initial relevance selection. A few representative examples of such relevance selection strategies will be discussed.

Work in [22] used the “Google Distance” to select URIs relevant to a query. The essential idea of their approach is to disregard the Semantic-Web-prescribed semantics of URIs. The prescribed semantics states that URIs are logical identifiers without any intrinsic meaning.

However, in practice, they are often made up of meaningful strings or compositions of natural language words. Huang et al. took the freedom to regard URIs as if they were natural language terms, and then used distance measures between words on the regular (non-semantic) Web of text as a proxy for a semantic distance measure between the URIs. The Google Distance was used as the proxy distance measure to be used on the regular Web. This Google distance (originally defined [9]) can be understood intuitively as the symmetric conditional probability of co-occurrence on the Web of two given terms and can be taken as a measure for their semantic relatedness. Benchmarks show the remarkable accuracy of this approach: Within only very small Google distances, sufficient triples were retrieved from large graphs to calculate meaningful answers to benchmark queries.

Even more remarkable is the earlier work by the same authors, in [23], which shows that even trivial and purely syntactic distance measures (retrieving only those URLs that are at some distance k from a given URL, e.g., taken from the query) give almost complete recall for benchmark queries, even for very small values of k .

A good explanatory theory for this effect is still missing, but apparently realistic graphs on the Semantic Web have sufficiently strong locality properties for such local selection strategies to essentially enable inferencing to scale to Web size without having to treat triple sizes of the order of the entire Web for any particular query.

More recent work by Williams et al. [42] on “scalable reduction” takes a different although related approach: A large dataset is reduced to a subset that is relevant to a particular type of query (e.g., queries related to people mentioned in the dataset). This reduced dataset is then only usable for queries of the specified type, but can be processed far more efficiently than the original full dataset.

The work presented in [14] deals with selecting data sources for the distributed evaluation of SPARQL queries. To this end, for each source, a data summary is created, consisting of an approximation of the locally stored triples. These summaries are used to estimate whether a source contains relevant information to the query and to rank information sources.

Hartig et al. [16] use the HTTP resolution protocol for selecting data sources. The triple URIs are resolved and the relevant data are retrieved. The obvious advantage of this method is that it does not require any additional infrastructure. Nevertheless, it is limited to URIs that can be resolved.

Unpublished work by Peikov et al. [48] uses classical activation algorithms to select the nodes of an RDF graph that are relevant to answering a query. So, in contrast with the work by Williams et al., and similar to the work by Huang et al., the graph selection takes place for every query (instead of one for a whole family of queries, as by Williams et al.). Evaluation of this work still remains to be published at the time of writing this chapter.

11.1.2.2 Limiting the Scope of Statements

For web-scale reasoning, a closed-world assumption, where information is controlled and trusted, is not realistic. Compared to the traditional Web, an additional challenge

is posed on the Semantic Web: In the traditional Web, erroneous information or malicious information providers can have an adverse effect on the system by causing information overload or spamming; in any case, their effect can be contained by humans recognizing the erroneous information. On the Semantic Web, the human is not always in the loop, so recognizing malicious information providers is far more difficult. To make matters worse, erroneous ontological information can lead to erroneous inferences that are not contained in the context in which the erroneous information was provided.

The work in SAOR [21] introduces the notion of “authoritative sources” for Semantic Web statements. The motivation lies in preventing the so-called ontology hijacking, where new ontologies redefine the semantics of existing entities in other ontologies.

Ontology hijacking can lead to very undesirable behavior concerning scalability. As an example, in [45], it is reported that applying full RDFS materialization on 78 million triples harvested from Swoogle yields a closure of 1.5 billion triples, a 192-fold increase. In comparison, the closure of one billion triples, using the SAOR system and OWL Horst semantics was three billion triples, a threefold increase.

11.2 Example Applications

Web-scale reasoning is mainly relevant in the context of Semantic Web Repositories and Semantic Web Search Engines. Since these two issues are dealt separately in [▶ Storing the Semantic Web: Repositories](#) and [▶ Semantic Web Search Engines](#), emphasis will be given on the scale aspect.

A number of large Semantic Web corpora encompassing several datasets each have been put together. FactForge consists of general knowledge gathered from DBpedia, Geonames, WordNet, and Freebase, among others. It consists of 1.3 billion explicit statements (<http://factforge.net/>) and 8.5 billion implicit statements. LinkedLifeData (<http://linkedlifedata.com>) and Bio2RDF (<http://bio2rdf.org>) consist of biomedical data. LinkedLifeData is represented by around 4 billion triples and Bio2RDF by 22 billion. Fairly expressive (RDFS or OWL Horst) reasoning can be performed on these datasets, and they can be queried using SPARQL. For more details, the reader is referred to [▶ Storing the Semantic Web: Repositories](#). It is interesting to note that while these datasets are web-scale in terms of size, they do not suffer from the quality problems outlined in [21].

In contrast, Semantic Web Search Engines deal with large amounts of poor quality data crawled from the Web. To achieve this, they compromise in terms of query and reasoning expressivity, typically supporting keyword queries and retrieval based on statement patterns, instead of full SPARQL. Sindice [33] uses a cluster infrastructure based on Hadoop and HBase for data processing [32]. As reported in [11], some reasoning is supported by means of context-dependent RDFS and partial OWL inference based on a persistent TBox composed of a network of Web ontologies.

11.3 Related Resources

In this section, a more thorough description of some of the approaches for web-scale reasoning presented in [Sect. 11.1](#) is given.

11.3.1 MaRVIN

MaRVIN [26, 35] performs materialization using a peer-to-peer architecture on a cluster infrastructure. It is motivated by the observation that it is hard to solve Semantic Web problems through traditional divide-and-conquer strategies since Semantic Web data are hard to partition. MaRVIN brings forward a method named divide-conquer-swap to do inferencing through forward chaining (i.e., calculate the closure of the input data). The main algorithm can be described in the following steps (Algorithm 2): First, the platform divides the input data into several independent partitions and assigns these partitions to compute nodes. Second, each node computes the closure of its partition using a conventional reasoner. Then, old and new data are mixed and new partitions are created in a distributed manner. This process is repeated until no new triples are derived. At this point, the full closure has been calculated.

The advantages of the MaRVIN platform are the following:

- Since the partitions are of equal size, the amount of data to be stored and the computation to be performed are evenly distributed among nodes.
- No upfront data analysis is required.
- It can support any monotonic logic by changing the reasoner.
- It uses a peer-to-peer architecture with no central coordination.
- It shows anytime behavior, that is, it produces results incrementally with time.

Algorithm 2 Divide-Conquer-Swap

1. The input data are divided into smaller partitions, which are stored on a shared location.
2. A number of reasoners are started on several computational nodes. Each node reads some input partitions and computes the corresponding output of this input data at its own speed.
3. On completion, each node selects some parts of the computed data and the input data, and sends them to some other node(s) for further processing.
4. Each node copies (parts of) the computed data to some external storage where the data can be queried on behalf of end users. These results grow gradually over time, producing anytime behavior.

The main disadvantage of MaRVIN is that it is slower than systems partitioning data according to the rule-set (such as DHTs or the approaches in [45, 46]).

Experiments on real-world datasets show that MaRVIN can calculate the closure of 200 M triples on 64 compute nodes in 7.2 min, yielding a throughput of 450 K triples/s [26].

11.3.2 WebPIE

WebPIE (Web-scale Parallel Inference Engine) [44, 45] is a high-performance OWL inferencing engine. It is built using the MapReduce [10] parallel programming model and the Hadoop framework. MapReduce requires that all information be encoded as sets of key-value pairs. These values must be processed using two functions: a map function, which transforms a pair of values to a set of new pairs, and a reduce function, which processes all key-value pairs with a given key to produce a new set of key-value pairs.

Using a mechanism similar to SPO-hashing, terms can be used as keys in order to bring triples that share terms to the same reduce function. Then, it is possible to apply reasoning locally. In the basic scheme, this process has to be repeated until fixpoint.

It was shown that this implementation performs poorly (for similar reasons as in DHT-based reasoning). To support efficient reasoning, the following methods and optimizations were used:

- A method for the parallel compression of RDF data, as described in [43]. This method was used to vastly decrease the size of the intermediate results.
- Schema triples were loaded in memory (similar to the approach in [6]).
- The map function was used to group together triples that could lead to the same triple being derived multiple times.
- The application of RDFS rules was ordered so as to minimize the need for fixpoint iteration.
- For rules with multiple joins in the antecedent, the small side of the join was loaded in memory.
- Rules that could repeatedly derive the same conclusions (e.g., rules involving the “owl:sameAs” predicate) were applied in only part of the graph at a given time.

An implementation on top of the Hadoop platform (<http://hadoop.apache.org/>) took 1 h to compute the RDFS closure of one billion triples on 33 machines. The throughput yielded was 123,000 triples/s in the input and 4.27 million triples/s in the output. This result was improved upon in [44] in two ways: The inferential expressivity was increased to cover the OWL Horst fragment [41], and the scale was stretched to 100 billion synthetic triples from the LUBM benchmark (closure in 45 h on 64 machines) and 1.5 billion triples from UniProt (closure in 6 h on 32 machines).

11.3.3 Scalable Reduction

In [42], Williams et al. employ high-performance computing to extract “interesting subsets” from large, diverse datasets. The approach consists of four steps:

1. *Defining target ontology* The user begins by defining the domain of interest. This is achieved by providing a so-called target ontology. In their example, Williams et al.

define an upper ontology describing persons and information about persons (name, e-mail address, etc.), and this upper ontology is mapped to common ontologies that describe persons in order to effectively integrate the common ontologies. The combination of the upper and common ontologies form the target ontology for their example.

2. *Reasoning* The cluster-based approach from [46] is then used to perform RDFS reasoning. Unlike in [46], only a well-defined partial RDFS closure is computed (excluding literal generalization and rules 1, 4a, 4b, 6, 8, and 10 from [Table 11.1](#)), and the closure is complete only with respect to the target ontology (there may be other ontologies in the dataset for which sound but incomplete inferences are produced). This process took about 30 min on 1,788 processors and produced 1.62 billion triples (including the original dataset of 899 million triples).
3. *Query/extraction* The data from the reasoning step (i.e., the original dataset plus inferences) are then queried using a SPARQL CONSTRUCT query to extract only specific data of interest. The parallel-hash-join-based approach by Weaver and Williams [53] was used on a BlueGene/L supercomputer to extract persons with both full names and e-mail addresses in 940 s using 8,192 nodes. The extracted dataset was approximately 785,000 triples.
4. *Compression* In a final effort to reduce the size, the extracted dataset from the previous step is then converted into a BitMat [53], which can be queried or analyzed more specifically. Constructing the BitMat took 25 s on a typical laptop.

11.3.4 Cray XMT

In [54], a high-performance RDFS reasoning method for the Cray XMT supercomputer is described. The Cray XMT is a shared memory multiprocessor machine, with very large memory configurations. The data flow of this system is similar to the one in [44] for RDFS reasoning: Schema triples are replicated to the local memory of each node and fixpoint iteration is minimized by ordering the RDFS rules. The most significant difference between the two systems is that while [44] uses MapReduce primitives to distribute data and eliminate duplicate derivations, this system uses a hashtable kept in shared memory. Experiments are reported using a Cray XMT setup with 512 processors and 4 TB of shared memory. For the LUBM benchmark with 5 billion triples, the obtained results were very promising, yielding a reasoning rate of 13.2 million triples/s. It is worth noting that the Cray MT performs all reasoning in main memory, unlike the other approaches in this section. The LarKC platform <http://www.larkc.eu> enables the construction of high-performance Semantic Web workflows. It provides an infrastructure with which developers can create high-performance applications plus a set of methods for scalable reasoning. Furthermore, dealing with the vastness of Web data, LarKC focuses on the trade-off between performance and completeness and correctness.

11.3.5 LarKC

Some key features of the LarKC platform are the following:

- A LarKC workflow consists of plug-ins, which are user-defined and can be shared and reused. Such workflows are automatically configured by the platform on the basis of an RDF description of the plug-ins and their connections.
- Plug-ins can communicate in different ways: blocking and nonblocking communication are supported, streaming behavior is supported, and input streams can be split across multiple plug-ins and output can be merged again.
- It is possible to wrap legacy applications and Web Services as plug-ins. Both standard DL reasoners and nonstandard reasoners (e.g., WebPIE), as well as indexing services such as Sindice <http://www.sindice.com>, have been successfully wrapped as plug-ins.
- The LarKC platform provides support for a set of well-known parallelization techniques, such as MapReduce, MPI, and multithreading.
- LarKC supports automatic parallelization of plug-ins when they operate on independent parts of data.
- The LarKC platform provides support for remote execution of plug-ins, enabling a workflow to run some plug-ins as remote Web Services or to run some compute-intensive workflows on dedicated hardware such as a compute cluster.
- A scalable data layer, based on the OWLIM repository ([▶ Storing the Semantic Web: Repositories](#)). Besides high performance, a crucial feature of the data layer is that it allows to pass data between plug-ins by reference, thereby minimizing communication overhead.
- The LarKC platform can interface with external services using multiple protocols. To this end, so-called endpoints are used, which are specified by the user, similar to plug-ins.
- A registry that lists available plug-ins. This set is directly coupled to a Web-based public “plug-in marketplace” where third parties can make their plug-ins available.

The functionality of the platform is provided by the following components:

- The LarKC Runtime Environment – serves as the “control center” of the platform, instantiating and invoking workflows
- The Management Interface – essentially the “user interface” of the platform
- The Plug-in Registry – a library and index of the available plug-ins
- The Data Layer – the storage facility for data and metadata in the platform

The LarKC platform has been used to build experimental workflows in cancer research (detecting genes that are likely to be relevant to certain types of cancer), in traffic-aware route planning in Milan, and in city administration (managing road-sign infrastructure in Seoul).

11.3.6 SAOR

The work in SAOR by Hogan et al. [21] deals with data quality and scalability for OWL reasoning over Web data in a centralized setting. The authors have developed a rule-based subset of OWL with the following characteristics:

- *Computational efficiency* achieved by, among others, the separation of terminological and assertional data.
- *Size reduction* of the produced inference, for example, by omitting less useful inferences based on the reflexivity of “owl:sameAs” and class membership for “rdfs:Resource.”
- *Protection against undesirable inferences* due to either erroneous or malicious statements. To achieve this, the notion of “authoritative sources” was introduced, denoting which sources are allowed to make what statements about URIs. Only statements from “authoritative sources” would be taken into consideration during reasoning.

A method for incomplete OWL reasoning was developed for this rule-set. Exploiting the separation of terminological and assertional data made possible by this rule-set, a combination of in-memory indexes, on-disk sorts, and file scans was used to achieve scalability. Experiments were performed with up to 1.1 billion statements on a single machine. SAOR has shown that forward-chaining materialization is feasible on Web data.

11.4 Future Issues

Future work toward web-scale reasoning will have to address at least the following central questions:

Centralization or distribution? Distribution is often claimed to be a promising road to scalability, and also one that meshes very well with the distributed nature of the Web. But surprisingly, centralization seems to have worked very well on today’s Web: The large search engines all work by locally caching the entire Web and running indexes, etc., on that local, centralized cache. Will distribution be needed for the Semantic Web?

Logic or IR? The world of logic is dictated by the discrete criteria of soundness and completeness: A calculus or an inference engine is either sound and complete, or it isn’t. Information retrieval on the other hand works with the measures of recall and precision, which can take on any value between 0 and 1. Until now, the Semantic Web has been dominated by the logical approach, but scalability might well be served by adopting the IR approach.

Forward or backward inference? Most available scalable RDF stores implement inference as forward reasoning, deriving all consequences from an RDF/OWL graph materializing the full closure. Very few scalable tools have explored the more traditional query-driven backward-reasoning processes.

Which parameters matter? Every time a new benchmarking study of Semantic Web stores or reasoners is published, a debate emerges about what the benchmarks should

vary and what parameters should be measured. In short, there is very little agreement on which parameters should be used to define the problem space of “scalability” on the Semantic Web.

Abandoning soundness and completeness Besides “simply” improving raw performance, a wide variety of strategies is potentially available to achieve higher performance at the price of dropping some of the classical requirements on sound and complete reasoning. Although these strategies are well known in a variety of subfields in AI, not many of them have been explored in the context of web-scale reasoning:

- Satisficing strategies
- Heuristics for caching
- Trading quality for time, soundness, or completeness
- Ranking strategies
- Anytime behavior
- Knowledge compilation
- Forgetting
- Statistical analysis of data and query patterns

Ranking answers Some combination of ranking and reasoning will be required if reasoners are to escape from the logical “fully complete” paradigm (and even within that paradigm, ranking of answers would be important). It would also require to recognize that statements that are logically equivalent do not always have to be treated in equivalent ways. It would be possible to do this either in a loose coupling (ranking applied to the results of a classical reasoner) or in a tight coupling (ranking built in to the reasoning algorithm). The source of such a ranking could be such extra-logical sources as the origin of the statements or the strings used for the identifiers occurring in the statements. One would also have to decide which items would be the subject of such rankings: query answers, triples, nodes, molecules, documents, or entire graphs?

Centralization With the advent of scalable reasoners, there is a tendency to centralize Semantic Web reasoning. As shown in the previous sections, pragmatic large-scale reasoning follows a model where data are harvested and stored in large repositories capable of inferencing over billions of triples. Encouraging results give the illusion that such computing infrastructures are a panacea for scalable reasoning on the Semantic Web.

It should be noted that this centralization comes at a price: (a) privacy is compromised, since the organization controlling the infrastructure has access to all information, (b) the central infrastructure has to be trusted by all participants, since it constitutes a single point of control, and (c) the functioning of the system relies solely on this infrastructure, centralizing cost and introducing a single point of failure.

It should also be noted that some distributed approaches made the above limitations worse. DHT-based reasoning performs even worse concerning privacy, since any node can store any information, severely compromising privacy. The same holds for trusting the infrastructure. Now, users have to trust all nodes instead of the central infrastructure.

Federated RDF stores is the only category of systems that can, in principle, deal with these limitations, since data are kept local to the publisher. Nevertheless, they do so at the expense of scalability, as described in [▶ Sect. 11.1.1.4](#).

A complicating factor in scalability is the potential dynamics of the datasets: The rate at which data are changing compared to the rate at which queries have to be answered, distinguishing between the frequency of data changes and the volumes of data that are involved in the changes. Dynamic datasets will become stale, hampering reuse of previous results and leading to incorrectness for such approaches as offline materialization and result-caching. An important special case is when the data are only changing monotonically, hence leading to incompleteness on stale datasets, but not to incorrectness. The severity of the results of staleness clearly depends on the use case for which the data and queries are deployed.

In the future, approaches that retain the distributed character of the Semantic Web while maintaining acceptable privacy, trust, dynamics, and scalability characteristics will become more relevant.

Benchmarking Current benchmarking was widely seen as inadequate, with the often-used LUBM and BSBM datasets seen as unrealistic and too regular (and hence too easy), and with the supposedly realistic Billion Triple Challenge datasets to be too incoherent. The LDSR dataset (<http://ldsr.ontotext.com>) promises to be a good alternative. Although the community has been fairly good at providing datasets, few if any of these datasets come with sets of queries to run in the benchmarks. Such benchmark queries were widely seen as badly needed. (Comment: since the Dagstuhl meeting, both the LDSR and the LLD datasets have been extended with a set of benchmark queries.) Finally, it was felt that the community should look at the experience in other communities as regards benchmarking, in particular communities such as databases, theorem proving, and information retrieval.

Evolutionary computing Technology for Semantic Web reasoners has been dominated by database-like approaches, where indexing and performing data joins efficiently are the core problems. In [34], an approach to perform SPARQL query answering using evolutionary computing is presented. The evolutionary RDF framework (eRDF) deals with very large amounts of RDF by applying evolutionary algorithms. Instead of the classical joins of partial solutions, full solutions to the query are guessed, checked for validity, and improved in an iterative process. At every stage of that process, the best-so-far result is made available. eRDF's evolutionary approach is specifically designed to find answers in an open world information space that is always changing. The current implementation of eRDF is used to power the similarity search service www.like.nu.

11.5 Cross-References

- ▶ KR and Reasoning on the Semantic Web: OWL
- ▶ Querying the Semantic Web: SPARQL

- Semantic Web Search Engines
- Storing the Semantic Web: Repositories
- Semantic Web Architecture

References

1. Adjiman, P., Chatalic, P., Goasdoué, F., Rousset, M.-C., Simon, L.: Distributed reasoning in a peer-to-peer setting: application to the semantic web. *J. Artif. Intell. Res.* **25**, 269–314 (2006)
2. Anadiotis, G., Kotoulas, S., Siebes, R.: An architecture for peer-to-peer reasoning. In: *New Forms of Reasoning for the Semantic Web: Scalable, Tolerant and Dynamic in the Sixth International Semantic Web Conference (ISWC 2007)*, Busan (2007)
3. Atre, M., Chaoji, V., Zaki, M.J., Hendler, J.A.: Matrix “Bit” loaded: a scalable lightweight join query processor for RDF data. In: Rappa, M., Jones, P., Freire, J., Chakrabarti, S. (eds.) *Proceedings of the 19th International World Wide Web Conference (WWW 2010)*, Raleigh, pp. 41–50. ACM, New York (2010)
4. Battré, D., Höing, A., Heine, F., Kao, O.: On triple dissemination, forward-chaining, and load balancing in DHT based RDF stores. In: *Fourth International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P) in Scope of 32nd International Conference on Very Large Data Bases (VLDB 2006)*, Seoul (2006)
5. Battré, D.: Query planning in DHT based RDF stores. In: *Proceedings of the 2008 IEEE International Conference on Signal Image Technology and Internet Based Systems (SITIS 2008)*. IEEE Computer Society, Washington, DC (2008)
6. Battré, D., Heine, F., Kao, O.: Top RDF query evaluation in structured P2P networks. In: Nagel, W.E., Walter, W.V., Lehner, W. (eds.) *Euro-Par. Lecture Notes in Computer Science*, vol. 4128, pp. 995–1004. Springer, Heidelberg (2006)
7. Byers, J., Considine, J., Mitzenmacher, M.: Simple load balancing for distributed hash tables. In: *Proceedings of the Second International Workshop on Peer-to-Peer Systems II (IPTPS 2003)*, Berkeley. *Lecture Notes in Computer Science*, vol. 2735, pp. 80–87. Springer, Heidelberg (2003)
8. Cai, M., Frank, M.: RDFPeers: a scalable distributed RDF repository based on a structured peer-to-peer network. In: *Proceedings of the 13th International World Wide Web Conference (WWW 2004)*, Manhattan. ACM, New York (2004)
9. Cilibrasi, R., Vitany, P.: The Google similarity distance. *IEEE/ACM Trans. Knowl. Data Eng.* **19**(3), 370–383 (2007)
10. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. In: *Symposium on Operating Systems Design and Implementation*, San Francisco, pp. 137–147 (2004)
11. Delbru, R., Polleres, A., Tummarello, G., Decker, S.: Context dependent reasoning for semantic documents in Sindice. In: *Proceedings of the Fourth International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2008)*, Karlsruhe (2008)
12. Ganguly, S., Silberschatz, A., Tsur, S.: A framework for the parallel processing of datalog queries. In: *Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data*, Atlantic City, pp. 143–152 (1990)
13. Haase, P., Broekstra, J., Ehrig, M., Menken, M., Mika, P., Plechawski, M., Pyszlak, P., Schnizler, B., Siebes, R., Staab, S., Tempich, C.: Bibster – a semantics-based bibliographic peer-to-peer system. In: *Proceedings of the Third International Semantic Web Conference (ISWC 2004)*, Hiroshima. *Lecture Notes in Computer Science*, vol. 3298, pp. 122–136. Springer, Berlin (2004)
14. Harth, A., Hose, K., Karnstedt, M., Polleres, A., Sattler, K., Umbrich, J.: Data summaries for on-demand queries over linked data. In: *Proceedings of the 19th International World Wide Web Conference (WWW 2010)*, Raleigh, pp. 411–420. ACM, New York (2010)
15. Harth, A., Umbrich, J., Hogan, A., Decker, S.: YARS2: a federated repository for querying graph structured data from the web. In: Aberer, K., Choi, K., Noy, N., Alle-mang, D., Lee, K., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D.,

- Schreiber, G., Cudre-Mauroux, P. (eds.) Proceedings of the Sixth International Semantic Web Conference and Second Asian Semantic Web Conference (ISWC/ASWC 2007), Busan. Lecture Notes in Computer Science, vol. 4825, pp. 211–224. Springer, Heidelberg (2007)
16. Hartig, O., Bizer, C., Freytag, J.-C.: Executing SPARQL queries over the web of linked data. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) Proceedings of the Eighth International Semantic Web Conference (ISWC 2009). Lecture Notes in Computer Science, vol. 5823, pp. 293–309. Springer, Heidelberg (2009)
 17. Hayes, P. (ed.) RDF semantics, W3C Recommendation (Feb 2004)
 18. Heimbigner, D., Mcleod, D.: A federated architecture for information management. *ACM Trans. Off. Inform. Syst.* **3**, 253–278 (1985)
 19. Heine, F.: Scalable P2P based RDF querying. In: Proceedings of the First International Conference on Scalable Information Systems (InfoScale 2006), p. 17. ACM, New York (2006)
 20. Heine, F., Hovestadt, M., Kao, O.: Processing complex RDF queries over P2P networks. In: Proceedings of the 2005 ACM Workshop on Information Retrieval in Peer-to-Peer Networks (P2PIR 2005), pp. 41–48. ACM, New York
 21. Hogan, A., Harth, A., Polleres, A.: Scalable authoritative OWL reasoning for the web. *Int. J. Semant. Web Inf. Syst.* **5**(2), 49–90 (2009)
 22. Huang, Z., van Harmelen, F.: Using semantic distances for reasoning with inconsistent ontologies. In: Proceedings of the Seventh International Semantic Web Conference (ISWC 2008), Karlsruhe. Lecture Notes in Computer Science, vol. 5318, pp. 178–194. Springer, Heidelberg (2008)
 23. Huang, Z., van Harmelen, F., ten Teije, A.: Reasoning with inconsistent ontologies. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005), Edinburgh, pp. 454–459 (2005)
 24. Idreos, S., Liarou, E., Koubarakis, M.: Continuous multi-way joins over distributed hash tables. In: Proceedings of the 11th International Conference on Extending Database Technology (EDBT 2008), pp. 594–605. ACM, New York (2008)
 25. Kaoudi, Z., Miliarakis, I., Koubarakis, M.: RDFS reasoning and query answering on top of DHTS. In: Proceedings of the Seventh International Semantic Web Conference (ISWC 2008), Karlsruhe. Lecture Notes in Computer Science, vol. 5318, pp. 499–516. Springer, Heidelberg (2008)
 26. Kotoulas, S., Oren, E., Harmelen, F.: Mind the data skew: distributed inferencing by speeddating in elastic regions. In: Rappa, M., Jones, P., Freire, J., Chakrabarti, S. (eds.) Proceedings of the 19th International World Wide Web Conference (WWW 2010), Raleigh, pp. 531–540. ACM, New York (2010)
 27. Liarou, E., Idreos, S., Koubarakis, M.: Evaluating conjunctive triple pattern queries over large structured overlay networks. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) Proceedings of the Fifth International Semantic Web Conference (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 399–413. Springer, Heidelberg (2006)
 28. Lua, K., Crowcroft, J., Pias, M., Sharma, R., Lim, S.: A survey and comparison of peer-to-peer overlay network schemes. *IEEE Commun. Surv. Tutorials* **7**(2), 72–93 (2004)
 29. Lum, V.Y.: Multi-attribute retrieval with combined indexes. *Commun. ACM* **13**(11), 660–665 (1970)
 30. Martínez, J.J., Cerdá, J.H.C.: A grid architecture for building hybrid museums. In: Chin-Wan Chung, Chong kwon Kim, Won Kim, Tok Wang Ling, and Kwan Ho Song (eds.) *Human. Society@Internet 2003*, Seoul. Lecture Notes in Computer Science, vol. 2713, pp. 312–322. Springer, Heidelberg (2003)
 31. Matono, A., Mirza, S., Kojima, I. (2006) RDFCube: A P2P-based three-dimensional index for structural joins on distributed triple stores. In: *Databases, Information Systems, and Peer-to-Peer Computing, Trondheim*. Lecture Notes in Computer Science, vol. 4125, pp. 323–330. Springer, Heidelberg (2006)
 32. Mika, P., Tummarello, G.: Web semantics in the clouds. *IEEE Intell. Syst.* **23**(5), 82–87 (2008)
 33. Oren, E., Delbru, R., Catasta, M., Cyganiak, R., Stenzhorn, H., Tummarello, G.: Sindice.com: a document-oriented lookup index for open linked data. *Int. J. Metadata Semant. Ontol.* **3**(1), 37–52 (2008)
 34. Oren E., Guéret, C., Schlobach, S.: Anytime query answering in RDF through evolutionary

- algorithms. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T.W., Thirunaryan, K. (eds.) Proceedings of the Seventh International Semantic Web Conference (ISWC 2008), Karlsruhe. Lecture Notes in Computer Science, vol. 5318, pp. 98–113. Springer, Heidelberg (2008)
35. Oren, E., Kotoulas, S., Anadiotis, G., Siebes, R., Ten Teije, A., Van Harmelen, F.: Marvin: Distributed reasoning over large-scale semantic web data. *J. Web Semant.* **7**(4), 305–316 (2009)
 36. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL web ontology language semantics and abstract syntax. Technical report, W3C Recommendation, University of Miami (Feb 2004)
 37. Serafini, L., Tamilin, A.: DRAGO: distributed reasoning architecture for the semantic web. In: Proceedings of the Second European Semantic Web Conference (ESWC 2005), Heraklion. Lecture Notes in Computer Science, vol. 3532, pp. 361–376. Springer, Berlin (2005)
 38. Sheth, A.P., Larson, J.A.: Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.* **22**, 183–236 (1990)
 39. Soma, R., Prasanna, V.K.: Parallel inferencing for OWL knowledge bases. In: Proceedings of the 37th International Conference on Parallel Processing (ICPP 2008), Portland, pp. 75–82 (2008)
 40. Stuckenschmidt, H., Vdovjak, R., Broekstra, J., Houben, G.-J.: Towards distributed processing of RDF path queries. *Int. J. Web Eng. Technol.* **2**(2–3), 207–230 (2005)
 41. ter Horst, H.J.: Completeness, decidability and complexity of entailment for RDF Schema and a semantic extension involving the OWL vocabulary. *J. Web Semant.* **3**(2–3), 79–115 (2005)
 42. Williams, G.T., Weaver, J., Atre, M., Hendler, J.A.: Scalable reduction of large datasets to interesting subsets. *J. Web Semant.* **8**(4), 365–373 (2010)
 43. Urbani, J., Maassen, J., Bal, H.: Massive semantic web data compression with MapReduce. In: Proceedings of the MapReduce Workshop at International Symposium of High Performance Distributed Computing (HPDC), Chicago (2010)
 44. Urbani, J., Kotoulas, S., Maassen, J., van Harmelen, F., Bal, H.: OWL reasoning with WebPie: calculating the closure of 100 billion triples. In: Proceedings of the Seventh European Semantic Web Conference (ESWC 2010), Heraklion. Lecture Notes in Computer Science, vol. 6088, pp. 213–227. Springer, Heidelberg (2010)
 45. Urbani, J., Kotoulas, S., Oren, E., van Harmelen, F. (2009) Scalable distributed reasoning using mapreduce. In: Proceedings of the Eighth International Semantic Web Conference (ISWC 2009), Chantilly. Lecture Notes in Computer Science, vol. 5823, pp. 634–649. Springer, Berlin (2009)
 46. Weaver, J., Hendler, J.: Parallel materialization of the finite RDFS closure for hundreds of millions of triples. In: Proceedings of the Eighth International Semantic Web Conference (ISWC 2009), Chantilly. Lecture Notes in Computer Science, vol. 5823, pp. 682–697. Springer, Berlin (2009)
 47. Wilkinson, B., Allen, M.: Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers, 2nd edn. Prentice-Hall, Upper Saddle River (2005)
 48. Peikov, I., et al.: Spreading activation components V2. LarKC deliverable D2.4.2. <http://www.larkc.eu/> (2010)
 49. Balduccini, M., Pontelli, E., Elkhatib, O., Le H.: Issues in parallel execution of non-monotonic reasoning systems. *Parallel Comput.* **31**(6), 608–647 (2005)
 50. Bonacina, M.P.: A taxonomy of parallel strategies for deduction. *Ann. Math. Artif. Intell.* **29**(1), 223–257 (2000)
 51. Gupta, G., Pontelli, E., Ali, K.A.M., Carlsson, M., Hermenegildo M.V.: Parallel execution of prolog programs: a survey. *ACM Trans. Program. Lang. Syst.* **23**, 472–602 (2001)
 52. Rudolph, S., Tserendorj, T., Hitzler, P.: What is approximate reasoning? In: Web Reasoning and Rule Systems Workshop, pp. 150–164 (2008)
 53. Weaver, J., Williams, G. T.: Scalable RDF query processing on clusters and supercomputers. In: Proceedings of the Fifth International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2009), Chantilly (2009)
 54. Goodman, E.L., Mizell, D.: Scalable in-memory RDFS closure on billions of triples. In: Proceedings of the Sixth International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2010) at the Ninth International Semantic Web Conference (ISWC 2010), Shanghai (2010)

12 Social Semantic Web

John G. Breslin^{1,2} · Alexandre Passant¹ · Denny Vrandečić³

¹Digital Enterprise Research Institute (DERI), National University of Ireland, Galway, Ireland

²School of Engineering and Informatics, National University of Ireland, Galway, Ireland

³Universitaet Karlsruhe, Karlsruhe, Germany

12.1	<i>Scientific and Technical Overview</i>	468
12.1.1	The Social Web	468
12.1.2	Issues with the Social Web	470
12.1.3	Bridging Social Web and Semantic Web Technologies	471
12.1.4	Ontologies for the Social Web	474
12.2	<i>Example Applications</i>	482
12.2.1	Semantic Blogging	482
12.2.2	Semantic Microblogging	486
12.2.3	Semantic Wikis	489
12.2.4	Semantic Social Bookmarking	493
12.2.5	Review Websites	496
12.2.6	Social Semantic Web Applications for Sharing Scientific Research	498
12.3	<i>Future Issues</i>	499
12.3.1	Searching the Social Semantic Web	499
12.3.2	Trust and Privacy on the Social Semantic Web	500
12.3.3	Integrating with the Social Semantic Desktop	501
12.4	<i>Cross-References</i>	502

Abstract: The Social Web has captured the attention of millions of users as well as billions of dollars in investment and acquisition. As more social websites form around the connections between people and their objects of interest, and as these “object-centered networks” grow bigger and more diverse, more intuitive methods are needed for representing and navigating content both within and across social websites. Also, to better enable user access to multiple sites and ultimately to content-creation facilities on the Web, interoperability among social websites is required in terms of both the content objects and the person-to-person networks expressed on each site. Semantic Web representation mechanisms are ideally suited to describing people and the objects that link them together, recording and representing the heterogeneous ties involved. The Semantic Web is also a useful platform for performing operations on diverse, distributed person- and object-related data. In the other direction, object-centered networks and user-centric services for generating collaborative content can serve as rich data sources for Semantic Web applications. This chapter will give an overview of the “Social Semantic Web,” where semantic technologies are being leveraged to overcome the aforementioned limitations in a variety of Social Web application areas.

12.1 Scientific and Technical Overview

12.1.1 The Social Web

A visible trend on the Web is the emergence of what is termed “Web 2.0” [97], a perceived second generation of Web-based communities and hosted services.

Although the term suggests a new version of the Web, it does not refer to an update of the World Wide Web’s technical specifications or its implementation, but rather to new structures and abstractions that have emerged on top of the traditional Web. Web 2.0 technologies, as defined in [4] and exemplified by sites such as *Wikipedia* [151], *Delicious* [28], *Flickr* [37] and *HousingMaps* [59], augment the Web and allow for an easier distributed collaboration. It is notable that the term Web 2.0 was actually not introduced to refer to a vision, but to characterize the current state of the art in Web engineering [97]. These technologies are distinguished from classical Web technologies by various characteristic features:

- *Community*: Web 2.0 pages allow contributors to collaborate and share information easily. The emerging result takes advantage of the *wisdom of the crowds* and could not have been achieved by each individual contributor, be it a music database like *freedb* [44] or an event calendar like *upcoming* [147]. Each contributor gains more from the system than they put into it.
- *Mashups*: Certain services from different sites can be pulled together in order to experience the data in a novel and enhanced way. This covers a whole range of handcrafted solutions, ranging from the dynamic embedding of AdSense advertisements [2] to the

visualization of Craigslist's housing information [21] on Google Maps [50], as realised by HousingMaps [59].

- *AJAX*: While existing prior to Web 2.0, *AJAX* – Asynchronous JavaScript + XML – is probably the technological pillar of Web 2.0 and allows the creation of responsive user interfaces, and thus facilitated both of the other pillars: community pages with slick user interfaces that could reach much wider audiences, and mash-ups that incorporate data from different websites, thereby introducing asynchronous communication for more responsive pages.

Social networking sites (SNSs) such as *Facebook* (one of the world's most popular SNSs [33]), *Friendster* (an early SNS previously popular in the USA, now widely used in Asia [46]), *orkut* (Google's SNS, popular in India and Brazil [99]), *LinkedIn* (an SNS for professional relationships [70]) and *MySpace* (a music and youth-oriented service [88]) – where explicitly stated networks of friendship form a core part of the website – have become part of the daily lives of millions of users, and have generated huge amounts of investment since they began to appear around 2002. Since then, the popularity of these sites has grown hugely and continues to do so.

Web 2.0 content-sharing sites with social networking functionality such as *YouTube* (a video-sharing site [158]), *Flickr* (for sharing images [37]) and *Last.fm* (a music community site [69]) have enjoyed similar popularity. The basic features of a social networking site are profiles, friend listings, and commenting, often along with other features such as private messaging, discussion forums, blogging, and media uploading and sharing. In addition to SNSs, other forms of social websites include wikis, forums, and blogs. Some of these publish content in structured formats enabling them to be aggregated together.

A common property of Web 2.0 technologies is that they facilitate collaboration and sharing between users with low technical barriers although usually on single sites and with a limited range of information. In this chapter, the term “Social Web” will be used to refer to this collaborative and sharing aspect, a term that can be used to describe a subset of Web interactions that are highly social, conversational, and participatory. The Social Web describes the collaborative part of the Web 2.0.

The Social Web has applications on intranets as well as on the Internet. Within companies and other organizations, Enterprise 2.0 [75] applications (i.e., Web 2.0-type tools deployed within an intranet) are being used for knowledge management, collaboration, and communication between employees. Some companies are also trying to make social website users part of their IT “team,” by allowing users to have access to some of their data and by bringing the results into their business processes [139]. Often the overly optimistic introduction of Social Web applications to a company suffers from a number of fallacies, like that Web 2.0 applications are often successful (mostly they are not), and that what works on the Web will work in an enterprise [31]. On the Web, applications of the Social Web are generally leisure oriented. People share pictures (*Flickr* [37]), videos (*YouTube* [158]), bookmarks (*Delicious* [28]), etc. However, serious

usage of Web 2.0 is now emerging more and more often. Social networks of researchers such as the *Nature Networks* [89] as well as bibliography management services such as *Bibsonomy* [11] have appeared on the Web, so that the collaborative aspects of Web 2.0 can be used as a better means toward integrating discussions and collaboration in scientific communities.

12.1.2 Issues with the Social Web

The Social Web is allowing people to connect and communicate via the Internet, resulting in the creation of shared, interactive spaces for communities and collaboration. There is currently a large disconnect in the online social space. A limitation of current social websites is that they are isolated from one another like islands in the sea, acting as closed-world and independent data silos. Different social websites can contain complementary knowledge and topics – segmented parts of an answer or solution that a person may be looking for – but the people participating in one website do not have ready access to relevant information available from other places. For instance, someone looking for information about a particular product, for example, a new camera, may find lots of comments about it on *Twitter*, pictures on *Flickr*, and videos on *YouTube*, but cannot get a view of all these information weighted by their own social network. As more and more social websites, communities, and services come online, the lack of interoperability between them becomes obvious. The Social Web creates a set of single data silos that cannot interoperate with each other, where synergies are expensive to exploit, and where reuse and interlinking of data is difficult and cumbersome.

A major reason for many of these social networks to remain walled gardens is that a network can thus bind users to its website. Since a user cannot move from one social network to the other without losing the previously added and maintained information, accepted connections, and history, a social network can lock in its users by not providing interoperability. Metcalfe's Law states that *the value of a telecommunications network is proportional to the square of the number of connected users of the system* [152]. In their paper [57], Hendler and Golbeck make an analogy between this law and the Social Web, and in particular discuss how Semantic Web technologies can be used to create bridges between social data and therefore enhance the global value of the network. In this way, the more that people interact with each other, the more knowledge becomes interlinked on a global scale. But this increase in value is not necessarily reflected in the relative value of current social network providers against each other, that is, the relative value of a major social network provider like *Facebook* [33] compared to its competitors may not necessarily increase by opening up its data, although the global value of the network and the value for each user will increase. As well as making its pages more openly accessible, Facebook is also making data available using the Open Graph Protocol [93], although this may be for economic reasons (increased search and advertising potential) rather than for the benefit of users (enabling personal profile portability).

A more technical reason for the lack of interoperation is that for most Social Web applications, communities, and domains, there are still no common standards for knowledge and information exchange or interoperation available. RSS (Really Simple Syndication or RDF Site Syndication, depending on the version used), a format for indicating recently updated Web content such as blog entries, was the first step toward interoperability among social websites, but it has various limitations that make it difficult for it to be used efficiently in such an interoperability context.

Blogs, forums, wikis, and social networking sites all can contain vibrant active communities, but it is difficult to reuse and to identify common data across these sites. For example, *Wikipedia* contains a huge body of publicly accessible knowledge, but reuse of this knowledge outside of Wikipedia and incorporating it into other applications poses a significant challenge. As another example, a user may create content on several blogs, wikis, and forums, but one cannot identify this user's contribution across all the different types of social software sites.

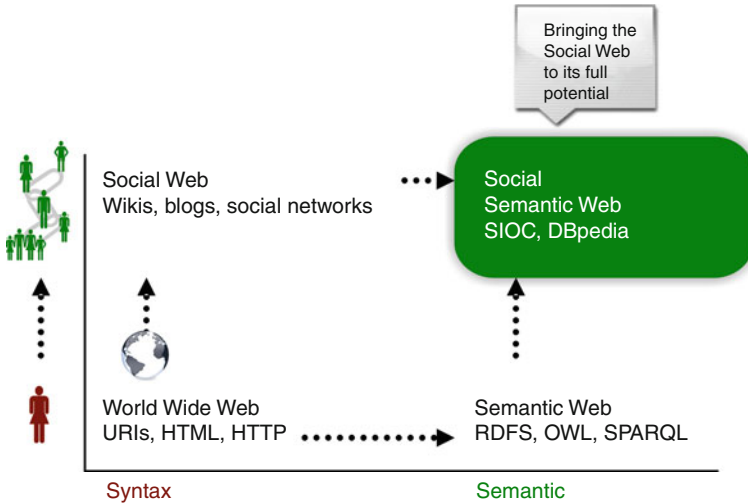
On the Web, navigation of data across social websites can be a major challenge. Communities are often dispersed across numerous different sites and platforms. For example, a group of people interested in a particular topic may share photos on *Flickr*, bookmarks on *del.icio.us* and hold conversations on a discussion forum. Additionally, a single person may hold several separate online accounts, and have a different network of friends on each. The information existing on each of these websites is generally disconnected, lacking in exchangeable semantics, and is centrally controlled by a single organization. Individuals generally lack control or ownership of their own data. Social websites are becoming more prevalent and content is more distributed. This presents new challenges for navigating such data.

12.1.3 Bridging Social Web and Semantic Web Technologies

The Semantic Web aims to provide the tools that are necessary to define extensible and flexible standards for information exchange and interoperability.

A number of Semantic Web vocabularies have achieved wide deployment: successful examples include RSS 1.0 for the syndication of information (RSS 1.0 being an RDF format, contrary to other RSS versions [112]), FOAF (Friend of a Friend [16, 39]) for expressing personal profile and social networking information, and SIOC (Semantically Interlinked Online Communities [14, 121]) for interlinking communities and distributed conversations.

The Semantic Web effort is in an ideal position to make social websites interoperable by providing standards to support data interchange and interoperation between applications, enabling individuals and communities to participate in the creation of distributed interoperable information. The application of the Semantic Web to the Social Web is leading to the “Social Semantic Web,” creating a network of interlinked and semantically rich knowledge, bringing together applications and social features of the Social Web with knowledge representation languages and formats from the Semantic Web (► Fig. 12.1).



■ Fig. 12.1

The Social Semantic Web

This vision of the Web will consist of interlinked documents, data, and even applications created by the end users themselves as the result of various social interactions, and it is described using machine-readable formats so that it can be used for purposes that the current state of the Social Web cannot achieve without difficulty. As Tim Berners-Lee said in a 2005 podcast [63], Semantic Web technologies can support online communities even as “online communities [...] support Semantic Web data by being the sources of people voluntarily connecting things together.” Therefore, integration between the Semantic Web and the Social Web is twofold:

- On the one hand, some efforts focus on using Semantic Web technologies to model social data. With ontologies such as FOAF and SIOC, social data can be represented using shared and common models, and therefore it becomes more easily interoperable and portable between applications.
- On the other hand, leveraging the wisdom of the crowds in Web 2.0 services can give a head start toward creating a large amount of Semantic Web data.

Additionally, the Social Web and social networking sites can contribute to the Semantic Web effort. Users of these sites often provide metadata in the form of annotations and tags on photos, ratings, blogroll links, etc. In this way, social networks and semantics can complement each other. Social website users are already creating extensive vocabularies and semantically rich annotations through folksonomies [79].

More than the sum of its parts: The combination of the Social Web and Semantic Web can lead to something greater than the sum of its parts: a Social Semantic Web where the islands of the Social Web can be interconnected with semantic technologies, and Semantic Web applications are enhanced with the wealth of knowledge inherent in user-generated content.

Because a consensus of community users is defining the meaning, these terms are serving as the objects around which those users form more tightly connected social networks. This goes hand-in-hand with solving the *chicken-and-egg* problem of the Semantic Web (i.e., you cannot create useful Semantic Web applications without the data to power them, and you cannot produce semantically rich data without the interesting applications themselves): since the Social Web contains such semantically rich content, interesting applications powered by Semantic Web technologies can be created immediately. The Social Semantic Web offers a number of possibilities in terms of increased automation and information dissemination that are not easily realizable with current social software applications:

- By providing a better interconnection of data, relevant information can be obtained from related social spaces (e.g., through social connections, inferred links, and other references).
- The Social Semantic Web would allow you to gather all your contributions and profiles across various sites (“subscribe to my brain,” similar to an enhanced version of *FriendFeed* [45]), or to gather and filter content from your friend/colleague connections.
- These semantically enhanced social spaces allow the use of the Web as a clipboard to allow exchange between various collaborative applications (e.g., by allowing readers to drag structured information from wiki pages into other applications, geographic data about locations on a wiki page could be used to annotate information on an event or a travel review in a blog post one is writing).
- Such interoperable social software applications can help users to avoid having to repeatedly express several times over the same information if they belong to different social spaces.
- New and innovative ways for personalizing the content and creating intelligent user interfaces for acquiring content can be created based on the growing amount of semantic information available about users, their interests, and relationships to other entities.
- These social spaces will also allow the creation of social semantic mash-ups, combining information from distributed data sources together that can also be enhanced with semantic information, for example, to provide the geolocation of someone’s friends in his social network who share similar interests with him.
- Fine-grained questions can be answered through such semantically enhanced social spaces, such as “show me all content by people both geographically and socially near to me on the topic of movies.”
- The Social Semantic Web can make use of emergent semantics to extract more information from both the content and any other embedded metadata.

There have been initial approaches in collaborative application areas to incorporate semantics in these applications with the aim of adding more functionality and enhancing data exchange – semantic wikis, semantic blogs, and semantic social networks (discussed in [Sect. 12.2](#)). These approaches require closer linkages and cross-application demonstrators to create further semantic integration both between and across application areas

(e.g., not just blog-to-blog connections, but also blog-to-wiki exchanges). A combination of such semantic functionality with existing grassroots efforts such as OpenID [95], a single sign-on mechanism, or OAuth [91], an authentication scheme, can bring the Social Web to another level. Not only will this lead to an increased number of enhanced applications, but an overall interconnected set of social software applications can be created using semantic technologies.

Some of the de facto standard ontologies that can be used for describing interlinked social spaces will now be described. It will follow with descriptions of a number of Social Semantic Web applications that have recently been developed. These use cases will serve as examples of how adding semantic information to social websites will enable richer applications to be built.

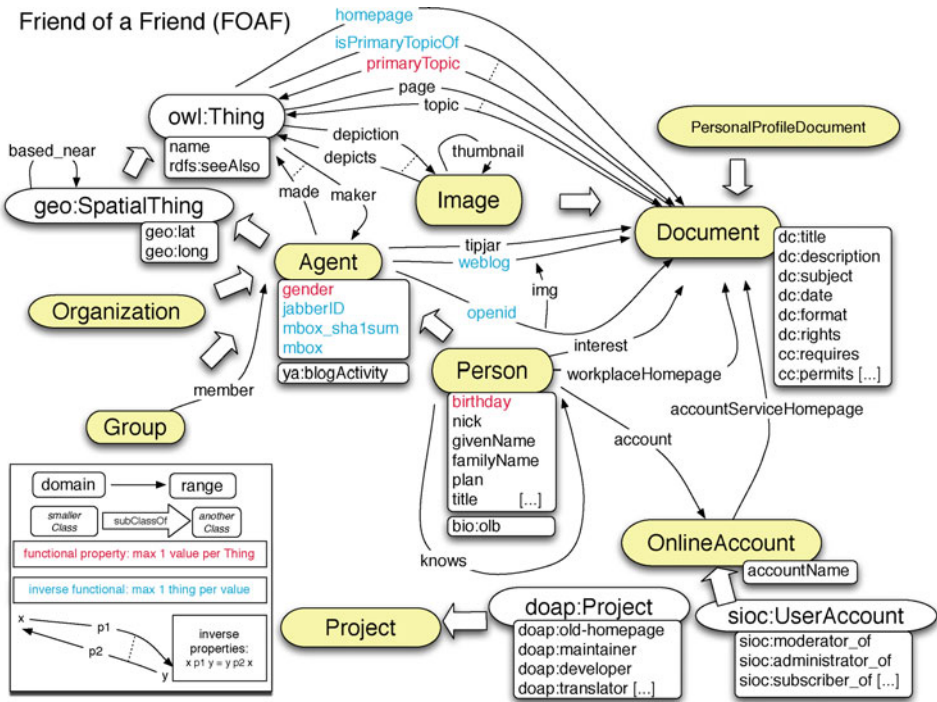
12.1.4 Ontologies for the Social Web

As discussed in [Ontologies and the Semantic Web](#), ontologies provide a shared model for representing semantically rich information on the Semantic Web. In addition, by using standard representation languages, such as RDF(S)/OWL, these ontologies can be shared across services, so that data become interoperable between distributed applications. In the realm of the Social Semantic Web, ontologies can be then used to represent uniformly the different artifacts produced and shared in social websites: communities, people, documents, tags, etc. In this section, some of the most popular ontologies for the Social Web are described.

FOAF – Friend of a Friend: The Friend-of-a-Friend (FOAF) project [39] was started by Dan Brickley and Libby Miller in 2000 and defines a widely used vocabulary for describing people and the relationships between them, as well as the things that they create and do. It enables people to create machine-readable Web pages for people, groups, organizations, and other related concepts. The main classes in the FOAF vocabulary ([Fig. 12.2](#), as illustrated by Dan Brickley [15]) include `foaf:Person` (for describing people), `foaf:OnlineAccount` (for detailing the online user accounts that they hold), and `foaf:Document` (for the documents that people create). Some of the most important properties are `foaf:knows` (used to create an acquaintance link), `foaf:mbox_sha1sum` (a hash over the eMail-address, often used as an identifier for a person and defined as an `owl:InverseFunctionalProperty` to allow smushing between instances), and `foaf:topic_interest` (used to point to resources representing an interest that a person may have).

foaf:knows is one of the most used FOAF properties: it acts as a simple way to create social networks through the addition of knows relationships for each individual that a person knows. For example, Bob may specify knows relationships for Alice and Caroline, and Damien may specify a knows relationship for Caroline and Eric; therefore Damien and Bob are connected indirectly via Caroline.

Anyone can create their own FOAF file describing themselves and their social network, using tools such as FOAF-a-matic [40] or FOAF Builder [41] from QDOS. In addition, the



■ Fig. 12.2

Friend-of-a-friend terms: Updated from Dan Brickley's CC picture (<http://www.flickr.com/photos/danbri/1855393361/>)

information from multiple FOAF files can easily be combined to obtain a higher-level view of the network across various sources. This means that a group of people can articulate their social network without the need for a single centralized database, following the distributed principles used in the architecture of the Web.

FOAF can be integrated with any other Semantic Web vocabularies, such as SIOC (described below) and SKOS – Simple Knowledge Organization System [123]. Some prominent social networking services that expose data using FOAF include *Hi5* (a social networking site [58]), *LiveJournal* (a social networking and blogging community site [71]), *Identi.ca* (a microblogging site [61]), and *MyBlogLog* (an application that adds community features to blogs [85]). People can also create their own FOAF document and link to it from their homepage. Aggregations of FOAF data from many individual homepages are creating distributed social networks; this can in turn be connected to FOAF data from larger online social networking sites. Third-party exporters are also available for major social websites including *Flickr* [38, 103], *Twitter* [146], *MySpace*, and *Facebook* [34, 111].

The knowledge representation of a person and their friends would be achieved through a FOAF fragment similar to that below.


```
@prefix foaf: <http://xmlns.com/foaf/0.1.>.
<http://www.johnbreslin.com/foaf/foaf.rdf#me> a foaf:Person;
  foaf:name "John Breslin";
  foaf:mbox <mailto:john.breslin@deri.org>;
  foaf:homepage <http://www.johnbreslin.com/>;
  foaf:nick "Cloud";
  foaf:depiction <http://www.johnbreslin.com/images/foaf_photo.jpg>;
  foaf:topic_interest <http://dbpedia.org/resource/SIOC>;
  foaf:knows [
    a foaf:Person;
    foaf:name "Sheila Kinsella";
    foaf:mbox <mailto:sheila.kinsella@deri.org>
  ];
  foaf:knows [
    a foaf:Person;
    foaf:name "Smitashree Choudhury";
    foaf:mbox <mailto:smitashree.choudhury@deri.org> ] .
```

hCard and XFN. hCard [55] is a microformat used to describe people, organizations, and contact details for both. It was devised by Tantek Çelik, and Brian.

Suda based on the vCard IETF format [148] for describing electronic business cards. Like FOAF, hCard can be used to define various properties relating to people, including “bday” (a person’s birth date), “email,” “nickname,” and “photo,” where these properties are embedded within XHTML attributes. The specification for hCard also incorporates the Geo microformat, which is used to identify the coordinates for a location or “adr” (address) described within an hCard. For example, the hCard for John Breslin is:

```
<div class="vcard">
<div class="fn">John Breslin</div>
<div class="nickname">Cloud</div>
<div class="org">National University of Ireland, Galway</div>
<div class="tel"> +35391492622</div>
<a class="url" href="http://johnbreslin.org/">
  http://johnbreslin.org/</a>
</div>
```

XFN (XHTML Friends Network) [154] is another social network-oriented microformat, developed by Tantek Çelik, Eric Meyer, and Matthew Mullenweg in 2003 just before the creation of the microformats community. XFN allows one to define relationships and relationship types between people, for example, “friend,” “neighbor,” “parent,” “met,” etc. XFN is also supported through the WordPress blogging platform: when adding a new blogroll link, one can use a form with checkboxes to specify additional metadata regarding the relationship between the blog owner and the person who is being

linked to (which is then exposed as metadata embedded in the blog's resulting XHTML). For example, an XFN "colleague"-type link to Uldis Bojārs would be written as:

```
<a href="http://captsolo.net" rel="colleague">Uldis Bojārs</a>
```

When combined with XFN, hCard provides similar functionality to FOAF in terms of describing people and their social networks. The different types of person-to-person relationships available in XFN allow richer descriptions of social networks to be created as FOAF voluntary only has a "knows" relationship. However, FOAF can also be extended with richer relationship types via the XFN in RDF vocabulary [155] developed in 2008 by Richard Cyganiak or the RELATIONSHIP vocabulary [108], which includes a variety of terms including `siblingOf`, `wouldLikeToKnow`, and `employerOf`.

SIOC – Semantically Interlinked Online Communities: SIOC aims to interlink related online community content from platforms such as blogs, message boards, and other social websites, by providing a lightweight ontology to describe the structure of and activities in online communities, as well as providing a complete food chain for such data. In combination with the FOAF vocabulary for describing people and their friends, and the SKOS model for organizing knowledge, SIOC lets developers link discussion posts and content items to other related discussions and items, people (via their associated user accounts), and topics (using specific "tags," hierarchical categories, or concepts represented with URIs).

As discussions begin to move beyond simple text-based conversations to include audio and video content, SIOC is evolving to describe not only conventional discussion platforms but also new Web-based communication and content-sharing mechanisms. At the moment, a lot of the content being created on social websites (events, bookmarks, videos, etc.) is being commented on and annotated by others. If you consider such content items to be the starting point for a discussion about the content (similar to a text-based thread starter in a forum or blog), and if the content item being created is done so in a container linked to a user or topic, then SIOC is quite suitable for describing metadata about these content items as well.

Since disconnected social websites require ontologies for interoperation, and due to the fact that there are a lot of social data with inherent semantics contained in these sites, there is potential for high impact through the successful deployment of a SIOC ontology. The development of SIOC was also an interesting process to explore how an ontology can be developed for and bootstrapped on the Semantic Web. Feedback from the research and development community to the ontology development process was increased through the development of a W3C Member Submission for SIOC [122].

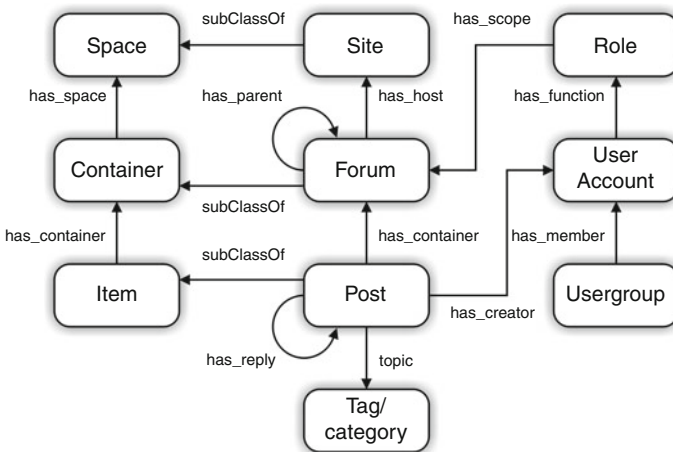
Many online communities still use mailing lists and message boards as their main communication mechanisms, and the SIOC initiative has also created a number of data producers for such systems in order to lift these communities to the Semantic Web. So far, SIOC has been adopted in a framework of about 60 applications or modules ranging from exporters for major Social Web platforms to applications in neuromedicine research, and has been deployed on hundreds of sites. Since the W3C Member Submission, SIOC has gained

even more success and attention from interested parties. For example, SIOC was recently chosen by Yahoo! SearchMonkey [116] as a suitable reference vocabulary to describe the activities of online communities, along with FOAF to describe the social networking stack.

An interesting aspect of SIOC is that it goes beyond pure Social Web systems and can be used in other use cases involving the need to model social interaction within communities, either in corporate environments (where there is a parallel lack of integration between social software and other systems in enterprise intranets), or for argumentative discussions and scientific discourse representation (e.g., via the SWAN/SIOC [135] initiative).

The ontology consists of the SIOC Core ontology, an RDF-based schema consisting of 11 classes and 53 properties, and 5 ontology modules: SIOC Access, SIOC Argumentation, SIOC Services, SIOC Types, and SWAN/SIOC. The SIOC Core ontology defines the main concepts and properties required to describe information from online communities on the Semantic Web. The main terms in the SIOC Core ontology are shown in [Fig. 12.3](#). The basic concepts in SIOC have been chosen to be as generic as possible, thereby allowing to describe many different kinds of user-generated content.

The SIOC Core ontology was originally created with the terms used to describe Web-based discussion areas such as blogs and message boards: namely Site, Forum, and Post. UserAccounts create Posts organized in Forums that are hosted on Sites. Posts have reply Posts, and Forums can be parents of other Forums. In parallel with the evolution of new types of social websites, these concepts became subclasses of higher-level concepts – data spaces (`sio:Space`), containers (`sio:Container`), and content items (`sio:Item`) – which were added to SIOC as it evolved. These classes allow one to structure the information in online community sites and distinguish between different kinds of objects. Properties defined in SIOC allow one to describe relations between objects and the attributes of these objects. For example:



■ Fig. 12.3

The SIOC ontology

- The `sioc:has_reply` property links reply posts to content that they are replying to.
- `sioc:has_creator` and `foaf:maker` link user-generated content to additional information about its authors.
- The `sioc:topic` property points to a resource describing the topic of content items, for example, their categories and tags.

The high-level concepts `sioc:Space`, `sioc:Container`, and `sioc:Item` are at the top of the SIOC class hierarchy, and most of the other SIOC classes are subclasses of these. A data space (`sioc:Space`) is a place where data reside, such as a website, personal desktop, shared file space, etc. It can be the location for a set of Container(s) of content Item(s). Subclasses of Container can be used to further specify typed groupings of Item(s) in online communities. The class `sioc:Item` is a high-level concept for content items and is used for describing user-created content. Usually these high-level concepts are used as abstract classes, from which other SIOC classes can be derived. They are needed to ensure that SIOC can evolve and be applied to specific domain areas where definitions of the original SIOC classes such as `sioc:Post` or `sioc:Forum` can be too narrow. For example, an address book, which describes a collection of social and professional contacts, is a type of `sioc:Container` but it is not the same as a discussion forum.

A sample instance of SIOC metadata from a forum (message board) in Drupal is shown below in Turtle. This forum has a title, a taxonomy topic in Drupal, a description, and is the container for one or more posts. More information on the posts can be obtained from the referenced URI (e.g., if it has replies, related posts, who wrote it, etc.).

```
<http://sioc-project.org/forum/13> a sioc:Forum;
  dc:title "Developers Forum" ;
  dc:description "Developers Forum at sioc-project.org" ;
  sioc:link <http://sioc-project.org/forum/13>;
  sioc:topic <http://sioc-project.org/taxonomy/term/13>;
  sioc:container_of <http://sioc-project.org/node/185>.
<http://sioc-project.org/node/185> a sioc:Post;
  rdfs:label "Microformats and SIOC" ;
  rdfs:seeAlso <http://sioc-project.org/sioc/node/185>.
```

A separate SIOC Types module defines more specific subclasses of the SIOC Core concepts, which can be used to describe the structure and various types of content of social websites. This module defines subtypes of SIOC objects needed for more precise representation of various elements of online community sites (e.g., `sioc_t:MessageBoard` is a subclass of `sioc:Forum`), and introduces new subclasses for describing different kinds of Social Web objects in SIOC. The module also points to existing ontologies suitable for describing full details of these objects (e.g., a `sioc_t:ReviewArea` may contain Review(s), described in detail using the Review Vocabulary). Examples of SIOC Core ontology classes and the corresponding SIOC Types module subclasses include: `sioc:Container` (`AddressBook`, `AnnotationSet`, `AudioChannel`, `BookmarkFolder`, `Briefcase`, `EventCalendar`, etc.); `sioc:Forum`: (`ChatChannel`, `MailingList`, `MessageBoard`, `Weblog`); and `sioc:Post` (`BlogPost`, `BoardPost`, `Comment`, `InstantMessage`, `MailMessage`,

WikiArticle). Some additional terms (Answer, BestAnswer, Question) were also added for question-and-answer sites like Yahoo! Answers [156], whereby content from such sites can also be lifted onto the Semantic Web.

Community sites typically publish Web service interfaces for programmatic search and content management services (typically SOAP and/or REST). These services may be generic in nature (with standardized signatures covering input and output message formats) or service specific (where service signatures are unique to specific functions performed, as can be seen in current Web 2.0 API usage patterns). The SIOC Services ontology module allows one to indicate that a Web service is associated with (located on) a `sioc:Site` or a part of it. This module provides a simple way to tell others about a Web service, and should not be confused with Web service definitions that define the details of a Web service. A `sioc_s:service_definition` property is used to relate a `sioc_s:Service` to its full Web service definition (e.g., in WSDL, the Web Services Description Language).

A SIOC Access module was created to define basic information about users' permissions, access rights, and the status of content items in online communities. User access rights are modeled using Roles assigned to a user and Permissions on content items associated with these Roles. This module includes terms like `sioc_a:Status` that can be assigned to content items to indicate their publication status (e.g., public, draft, etc.), and `sioc_a:Permission`, which describes a type of action that can be performed on an object (e.g., a `sioc:Forum`, `sioc:Site`) that is within the scope of a `sioc:Role`.

An argumentation module extension to SIOC has been provided to allow one to formulate agreement and disagreement between SIOC content items. The properties and classes defined in this SIOC Argument module are related to other argumentation models such as SALT [134] and IBIS [60]. Some related work has also been performed by aligning SIOC with the SWAN ontology for scientific discourse in neuromedicine in the SWAN/SIOC joint initiative [135], providing a common framework to model online conversations in these communities, from the item level to the conversational layer.

Ontologies for Semantic Tagging: While they have been wrongly opposed in some discussions, ontologies and free-tagging practices can be efficiently combined together. One approach to bridge folksonomies and the Semantic Web is to use RDF(S)/OWL modeling principles to represent tags, tagging actions, and other related objects such as tag clouds. While tag-based search is the only way to retrieve tagged content at the moment (leading to problems if people use different tags for the same meaning, or if they use the same tag for different meanings), these new models allow advanced querying capabilities such as “which items are tagged ‘semanticweb’ on any platform,” “what are the latest ten tags used by Denny on del.icio.us,” “list all the tags commonly used by Alex on SlideShare and by John on Flickr,” or “retrieve any content tagged with something relevant to the Semantic Web field.” Having tags and tagged content published in RDF also allows one to easily link this to or from other Semantic Web data, and to reuse it across applications in order to achieve the goal of a global graph of knowledge.

While it has not been implemented in RDF, Gruber [53] defined one of the first approaches to model folksonomies and tagging actions using a dedicated ontology. This work considers the tripartite model of tagging and extends it with (1) a space

attribute, aimed at modeling the website in which the tagging action occurred and (2) a polarity value in order to deal with spam issues. His proposal provides a complete model to represent tagging actions, but also considers the idea of a tag identity, such that various tags can refer to the same concept while being written differently, introducing the need to identify some common semantics in the tags themselves.

Mika discusses the emergent semantics of folksonomies and how they can be materialized through formal models, that is, ontologies [79]. In particular, he defines how ontologies can be mined from existing social bookmarking services by combining social network analysis with a tripartite model for representing ontologies, and therefore introduces a social component into ontology mining. The Tagora project [137] has also leveraged masses of data from collaborative tagging systems to examine the behavior of human agents on the Web and to help develop an understanding of the dynamics of information in online communities.

The Tag Ontology [138] was the first RDF-based model for representing tags and tagging actions, based on the initial ideas of Gruber and on the common theoretical model of tagging that was mentioned earlier. This ontology defines the “Tag” and “Tagging” classes with related properties to create the tripartite relationship of tagging. In order to represent the user involved in a tagging action, this ontology relies on the FOAF vocabulary. An important feature of this model is that it defines a Tag class, hence implying that each tag will have a proper URI so that tags can be used both as the subject and object of RDF triples. Since Tag is defined as a subclass of `skos:Concept` from the Simple Knowledge Organization System (SKOS), tags can be linked together, for example, to model that the “`rdfa`” tag is more specific than “`semanticweb`.”

The Social Semantic Cloud of Tags (SCOT) ontology [65, 115] is focused on representing tag clouds and defines ways to describe the use and co-occurrence of tags on a given social platform, allowing one to move his or her tags from one service to another and to share tag clouds with others. SCOT envisions data portability not for the content itself but for the tagging actions and the tags of a particular user. SCOT reuses the Tag Ontology as well as SIOC to model tags, tagging actions, and tag clouds. An important aspect of the SCOT model is that it considers the space where the tagging action happened (i.e., the social platform, e.g., Flickr or `del.icio.us`), as suggested by Gruber’s initial proposal. SCOT also provides various properties to define spelling variants between tags, using a main `spellingVariant` property and various subproperties such as `acronym`, `plural`, etc.

Finally, Meaning of a Tag (MOAT) [81] aims to represent the meaning of tags using URIs of existing domain ontology instances or resources from existing public knowledge bases [104], such as those from the Linking Open Data project (a set of best practice guidelines for publishing and interlinking pieces of data on the Semantic Web, see [▶ Semantic Annotation and Retrieval: Web of Data](#)). To solve issues with free-form tagging regarding information retrieval, MOAT allows us to model facts such as: In this blog post, Bob uses the tag “apple” and by that he means the computer brand identified by `dbpedia:Apple_Inc.`, while the “apple” tag on that other picture means the fruit identified by `dbpedia:Apple`. MOAT is more than a single model, as it also provides a framework [82] to let people easily bridge the gap between simple free-form tagging and semantic indexing, helping users to annotate their content with URIs of Semantic Web resources from

the tags that they have already used for annotated content. MOAT can also be automated as applied by [1] in the GroupMe! system.

More recently, the Common Tag initiative [19] (involving AdaptiveBlue, Faviki, Freebase, Yahoo!, Zemanta, Zigtag and DERI, NUI Galway) developed a lightweight vocabulary with a similar goal of linking tags to well-defined concepts (represented with their URIs) in order to make tagging more efficient and interconnected. In particular, it focuses on a simple approach allowing site owners to publish RDFa tag annotations, as well as providing a complete ecosystem of producers and consumers of Common Tag data that can help end users to deploy applications based on this format.

Other Ontologies: Various other models for modeling Social Web content have also been deployed. For example, in the wiki domain, WIF (Wiki Interchange Format) and WAF (Wiki Archive Format) have been developed [149] as common models to exchange and archive data between different wikis, as well as the WikiOnt vocabulary [54], with a more complete list of wiki-based models available [100].

Other application-specific models include SAM [42] and NABU [101] for instant messaging, as well as mle [107] and SWAML [36] for mailing list representation using Semantic Web technologies. These last two applications rely on SIOC, and the SWAML ontology has been completely integrated with SIOC.

In 2008, the Online Presence Ontology (OPO) [96] project was initiated for modeling online presence information. Whereas FOAF is mainly focused on static user profiles and SIOC has been somewhat oriented toward threaded discussions, OPO can be used to model dynamic aspects of a user's presence in the online world (e.g., custom messages, IM statuses, etc.). By expressing data using OPO, online presence data can be exchanged between services (chat platforms, social networks, and microblogging services). The ontology can also be used for exchanging IM statuses between IM platforms that use different status scales, since it enables very precise descriptions of IM statuses. The maintainers of OPO and SIOC are also working together to define alignments such that semantic descriptions of online presence and community-created content can be effectively leveraged on the Social Semantic Web.

APML [5], or Attention Profiling Markup Language, is an XML-based format that allows people to share their own personal "attention profile," similar to how OPML (Outline Processor Markup Language) allows the exchange of reading lists between sites and news readers. APML compresses all forms of attention-related data into a portable file format to provide a complete description of a person's rated interests (and dislikes). Efforts are also underway to link APML into the Semantic Web by creating an APML-RDF schema.

12.2 Example Applications

12.2.1 Semantic Blogging

A blog, or weblog, is a user-created website consisting of journal-style entries displayed in reverse-chronological order. Entries may contain text, links to other websites, and images

or other media. Often there is a facility for readers to leave comments on individual entries, which makes blogs a very interactive medium. The growth and take-up of blogs over the past 5 years has been dramatic, with a doubling in the size of the blogosphere every 6 months or so (according to statistics from Technorati [141]). Over 120,000 blogs are created every day, with nearly a million blog posts being made each day. Technorati counted 133 million blogs in 2008 [140]. Bloggers are often at the forefront of information, where traditional media cannot act as fast as the online “wisdom of crowds.”

Similar to accidentally wandering onto message boards and Web-enabled mailing lists, when searching for something on the Web, one often comes across a relevant entry on someone’s blog. RSS feeds are also a useful way of accessing information from your favorite blogs, but they are usually limited to the last 15 or 20 entries, and do not provide much information on exactly who wrote or commented on a particular post, or what the post is talking about. Some approaches like SIOC (outlined earlier) aim to enhance the semantic metadata provided about blogs, forums, and posts, but there is also a need for more information about what exactly a person is writing about. Blog entries often refer to resources on the Web and these resources will usually have a context in which they are being used, and in terms of which they could be described. For example, a post which critiques a particular resource could incorporate a rating, or a post announcing an event could include start and end times. When searching for particular information in or across blogs, it is often not that easy to get it because of “splogs” (spam blogs) and also because of the fact that the virtue of blogs so far has been their simplicity – apart from the subject field, everything and anything is stored in one big text field for content. Keyword searches may give some relevant results, but useful questions such as “find me all the Chinese restaurants that bloggers reviewed in Washington DC with a rating of at least 5 out of 10” cannot be posed, and you cannot easily drag-and-drop events or people or anything (apart from URLs) mentioned in blog posts into your own applications.

Blog posts are sometimes categorized (e.g., “Japan,” “Movies”) by the post creator using predefined categories or tags, such that those on similar topics can be grouped together using free-form tags/keywords or hierarchical tree categories. Posts can also be tagged by others using social bookmarking services like Delicious [28] or personal aggregators like Gregarius [52]. Other services like Technorati can then use these tags or keywords as category names for linking together blog posts, photos, links, etc. in order to build what they call a “tagged Web.” Using Semantic Web technology, both tags and hierarchical categorizations of blog posts can be further enriched and exposed in RDF via the SKOS framework.

Bloggng at present offers poor query possibilities (except for searching by keyword or seeing all posts labeled with a particular tag). Some linking of posts is possible via direct HTML links or trackbacks, but nothing can be said about the nature of those links (are you agreeing with someone, linking to an interesting post, or are you quoting someone whose blog post is directly in contradiction with your own opinions?). There have been some approaches to tackle the issue of adding more information to blog posts, such that: (1) advanced (more precise) queries can be made regarding the posts’ content; (2) the things that people talk about can be reused in other posts or applications (addresses,

events, etc.); and (3) “richer” links can be created between blog posts (going beyond the previously described techniques involving categories or tags). One approach is called structured blogging [133] (mainly using microformats to annotate blog content), and the other is semantic blogging (using RDF to represent both blog structures and blog content): both approaches can also be combined together.

Structured blogging is an open-source community effort that has created tools to provide microcontent (e.g., microformats or RDFa, see [▶ Semantic Annotation and Retrieval: Web of Hypertext – RDFa and Microformats](#)) from popular blogging platforms such as WordPress and Movable Type. In structured blogging, microcontent is positioned inline in the (X)HTML (and subsequent syndication feeds) and can be rendered via CSS. Although the original effort has tapered off, structured blogging is continuing through services like LouderVoice [73], a review site that integrates reviews written on blogs and other websites. In structured blogging, packages of structured data are becoming post components. Sometimes (not all of the time) a person will have a need for more structure in their posts – if they know a subject deeply, or if their observations or analyses recur in a similar manner throughout their blog – then they may best be served by filling in a form (which has its own metadata and model) during the post creation process. For example, someone may be writing a review of a film they went to see, or reporting on a sports game they attended, or creating a guide to tourist attractions they saw on their travels. Not only do people get to express themselves more clearly, but blogs can start to interoperate with enterprise applications through the microcontent that is being created in the background.

Semantic Blogging Applications: Semantic Web technologies can also be used to enhance any available post structures in a machine-readable way for more linkage and reuse, through various approaches in what is termed semantic blogging. Steve Cayzer [17] envisioned an initial idea for semantic blogging with two main aspects that could improve blogging platforms: a richer structure both for blog post metadata and their topics – using shared ontologies – and richer queries in terms of subscription, discovery, and navigation. He later defined a Snippet Manager service implementing some of these features. [64] gave some other ideas about “what would it mean to blog on the Semantic Web.” They argued that such tools should be able to produce structured and machine-understandable content in an autonomous way, without any additional input from the users. They also provided a first prototype based on the Haystack platform [106] that showed new ways to navigate between content thanks to these techniques. Traditional blogging is aimed at what can be called the “eyeball Web” – that is, text, images, or video content that is targeted mainly at people [83]. Semantic blogging aims to enrich traditional blogging with metadata about the structure (what relates to what and how) and the content (what is this post about – a person, event, book, etc.). Already RSS and Atom are used to describe blog entries in a machine-readable way and enable them to be aggregated together. However, by augmenting these data with additional structural and content-related metadata, new ways of querying and navigating blog data become possible.

It is not simply a matter of adding semantics for the sake of creating extra metadata, but rather a case of being able to reuse what data a person already has in their desktop or Web space and making the resulting metadata available to others. People are already

(sometimes unknowingly) collecting and creating large amounts of structured data on their computers, but these data are often tied into specific applications and locked within a user's desktop (e.g., contacts in a person's address book, events in a calendaring application, author and title information in documents, and audio metadata in MP3 files). Semantic blogging can be used to "lift" or release these data onto the Web (e.g., using applications like Shift [120]). For example, looking at the picture in [Fig. 12.4](#) [84], Andreas writes a blog post, which he annotates using content from his desktop address book application. He publishes this post on the Web, and someone else reading this post can reuse the embedded metadata in his or her own desktop applications (i.e., using the Web as a clipboard).

SparqlPress [127] is another prototype that leverages Semantic Web technologies in blogs. It is not a separate blogging system but rather an open-source plug-in for the popular WordPress platform [153], and it aims to produce, integrate, and reuse RDF data for an enhanced user experience. SparqlPress mainly relies on the FOAF, SIOC, and SKOS Semantic Web vocabularies. One interesting feature that SparqlPress provides is its linking of a FOAF social networking profile and an OpenID identity. This can be used to display extra information about the user on the blog, and it can also be useful for the blocking of blog comment spam. Finally, Zemanta [159] provides client-side and server-side tools that enrich the content being created by bloggers or publishers, allowing them to automatically add hyperlinks, choose appropriate tags (e.g., using the Common Tag framework), and insert images based on an analysis of the content being posted.

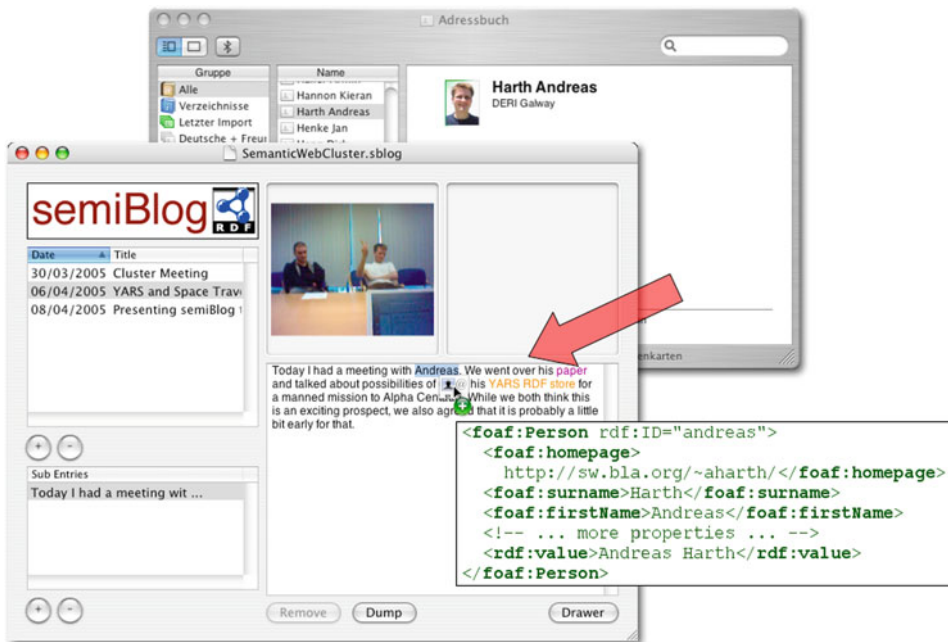


Fig. 12.4

Annotating a blog entry with an address book entry [83]

12.2.2 Semantic Microblogging

Microblogging is a recent social phenomenon on the Social Web, with similar usage motivations (i.e., personal expression and social connection) to other applications like blogging. It can be seen as a hybrid of blogging, instant messaging, and status notifications, allowing people to publish short messages (usually fewer than 140 characters) on the Web about what they are currently doing. These short messages, or microblog posts, are often called “tweets” (due to the most popular microblogging platform, Twitter [145]) and have a focus on real-time information. As a simple and agile form of communication in a fluid network of subscriptions, it offers new possibilities regarding lightweight information updates and exchange. Twitter is now one of the largest microblogging services, and the value of microblogging is demonstrated by its popularity and that of other services such as FriendFeed [45], StatusNet [129], etc. Some microblogging services also have SMS integration, allowing one to send updates and receive microblog posts from friends via a mobile phone.

Individuals can publish their brief text updates using various communication channels such as text messages from mobile phones, instant messaging, e-mail, and the Web. The simplicity of publishing such short updates in various situations or locations and the creation of a more flexible social network based on subscriptions and response posts makes microblogging an interesting communication method. Similar to how blogging led to “grassroots journalism,” microblogging has led to grassroots reporters, especially Twitter, as updates can be posted in many ways and from different devices (e.g., via text message from mobile phones). Hence, it was one of the first media to report the May 2008 Sichuan earthquake in China [9] and the November 2008 terrorist attacks in Mumbai [142].

Microblogging is quite useful for getting a snapshot of what is going on in and for interacting with your community or communities of interest. Similar to using a blog aggregator and scanning the titles and summaries of many blogs at once, thereby getting a feel for what is going on at a particular point in time, microblogging allows one to view status updates from many people in a compact (screen) space. If you are subscribed to a few hundred people it can be somewhat difficult to see all that is relevant since even the most interesting microbloggers will not be talking about stuff that is interesting to you all the time. However, Twitter clients like TweetDeck [143] do allow various searches to be set up in separate columns, such that updates relevant to a certain keyword or combination of keywords (e.g., “galway OR ireland,” “Semantic Web”) can be monitored quite easily, irrespective of whom one is following.

This communication method is also promising for corporate environments in facilitating informal communication, learning, and knowledge exchange (e.g., Yammer [157] is an enterprise microblogging platform). Its so far untapped potential can be compared to that of company-internal wikis some years ago. Microblogging can be characterized by rapid (almost real-time) knowledge exchange and fast propagation of new information. For a company, this can mean real-time questions-and-answers and improved informal learning and communication, as well as status notifications, for example, about upcoming meetings and deliveries. However, the potential for microblogging in corporate

environments still has to be demonstrated with real use cases (e.g., IBM has recently deployed an internal beta microblogging service called Blue Twit). It is expected that a trend of corporate microblogging will emerge in the next years similar to what happened with blogging, wikis, and other Enterprise 2.0 services (see [eScience](#)).

SMOB – Semantic MicrOBlogging: On the technology blog TechCrunch, Michael Arrington wrote a post [7] about the need for a “decentralised Twitter” via open alternative microblogging platforms, which was picked up by technologists Dave Winer, Marc Canter, and Chris Saad amongst others. The SMOB or semantic microblogging platform [125] developed in DERI, NUI Galway is an example of how Semantic Web technologies can provide an open platform for decentralized and distributed publishing of microblog content, mainly using the FOAF and SIOC vocabularies.

One aim of SMOB is to demonstrate how such technologies can provide users with a way to control, share, and remix their own data as they want to, not being solely dependent on the facilities provided by a third-party service, since SMOB-published data always belongs to the user who created it. As soon as someone writes some microblog content using a SMOB client, the content is spread through various microblogging servers or aggregators (including SMOB, Twitter, and Laconica), but remains available locally to the user who created it. Therefore, if one aggregator closes for some reason, the user can still use their local data somewhere else as it is primarily hosted by him or her and then aggregated by the third-party service.

In order to represent microblogging data, SMOB uses FOAF and SIOC to model microbloggers, their properties, account and service information, and the microblog updates that users create. A multitude of publishing services can ping one or a set of aggregating servers as selected by each user, and it is important to note that users retain control of their own data through self-hosting. The aggregate view of microblogs uses ARC [6] for storage and querying, and MIT’s Exhibit faceted browser [80] for the user interface as shown in [Fig. 12.5](#). It therefore offers a user-friendly interface for displaying complex RDF data aggregated from distributed sources. Furthermore, microblog posts can also embed semantic tags, for example, geographical tags, which can leverage the GeoNames database [47] to power new visualizations such as the map view in [Fig. 12.6](#). At the moment, the complete dataset of updates is publicly available and can be browsed using any RDF browser (Tabulator, Disco, etc.).

Other Initiatives: Other microblogging platforms leveraging semantics include smesher [124] and StatusNet [129] (formerly Laconica). smesher is a semantic microblogging client with local storage, that integrates with Twitter and Identi.ca (another popular microblogging website). It features structure identification and a dashboard for custom filters, and has a SPARQL API for querying (see [Querying the Semantic Web: SPARQL](#)). StatusNet, an open-source platform that powers Identi.ca, also publishes both FOAF (describing people) and SIOC data (as SIOC-augmented RSS feeds for users and groups). StatusNet also allows users to create friend connections across installations.

Related to semantic microblogging are various approaches for embedding semantics in microblog posts, including microturtle [78], the “Star Priority Notation” [128], and microsyntax [77].

Semantic MicroBlogging - demo timeline

BLOCS • LIGNE DE TEMPS • CARTE

24 MicroBlogPost

Trier par : [date](#), [puis par...](#) • Grouper selon le tri

Date
1 2008-02-05
19 2008-02-06
4 2008-03-14

Name
20 Alexandre Passant
4 Tuukka Hastrup

- 

Tuukka Hastrup
testing new client
2008-03-14T18:09:21+00:00
- 

Alexandre Passant
test
2008-03-14T03:26:19-07:00
- 

Alexandre Passant
one more test with new config file
2008-03-14T03:22:34-07:00
- 

Alexandre Passant
test
2008-03-14T03:19:16-07:00

Fig. 12.5

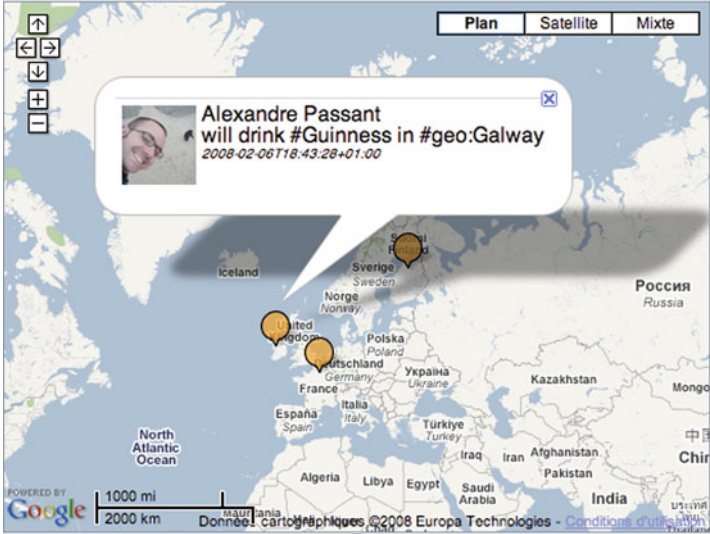
Latest SMOB updates rendered in Exhibit

Semantic MicroBlogging - demo timeline

BLOCS • LIGNE DE TEMPS • **CARTE**

24 MicroBlogPost

[21 résultats](#) sur 24 ne peuvent pas être tracés.



Date
1 2008-02-05
19 2008-02-06
4 2008-03-14

Name
20 Alexandre Passant
4 Tuukka Hastrup

Topic
22 (missing this field)
1 deri
1 foaf

Fig. 12.6

Map view of latest updates with Exhibit

12.2.3 Semantic Wikis

Many people are familiar with Wikipedia [151], but less know exactly what a wiki is. A wiki is a website that allows users to edit content through the same interface they use to browse it, usually a Web browser, while some desktop-based wikis also exist. This facilitates collaborative authoring in a community, especially since editing a wiki does not require advanced technical skills. A wiki consists of a set of Web pages which can be connected together by links. Users can create new pages (e.g., if one for a certain topic does not exist), and they can also change (or sometimes delete) existing ones, even those created by other members. The WikiWikiWeb was the first wiki, established by Ward Cunningham in March 1995, and the name is based on the Hawaiian term *wiki*, meaning “quick,” “fast,” or “to hasten.” Wikis often act as informational resources, like a reference manual, encyclopedia, or handbook. They amass to a group of Web pages where users can add content and others can edit the content, relying on cooperation, checks, and balances of its members, and a belief in the sharing of ideas. This creates a community effort in resource and information management, disseminating the “voice” amongst many instead of concentrating it upon few people. Therefore, contrary to how blogs reflect the opinions of a predefined set of writers (or a single author), wikis use an open approach whereby anyone can contribute to the value of the community.

Wikis are also being used for free dictionaries, book repositories, event organization, writing research papers, project proposals, and even software development or documentation. In this way, the openness of wiki-based writing can be seen as a natural follow-up to the openness of source-code modification. Wikis have become increasingly used in enterprise environments for collaborative purposes: research projects, papers and proposals, coordinating meetings, etc. SocialText [126] produced the first commercial open-source wiki solution, and many companies now use wikis as one of their main intranet collaboration tools. However, wikis may break some existing hierarchical barriers in organizations (due to a lack of workflow mechanisms, open editing by anyone with access, etc.), which means that new approaches toward information sharing must be taken into account when implementing wiki solutions.

There are hundreds of wiki software systems now available, ranging from MediaWiki [76], the software used on the Wikimedia family of sites, and Eu-gene Eric Kim’s PurpleWiki [105], where fine-grained elements on a wiki page are referenced by purple numbers (a concept from Doug Engelbart), to Alex Schröder’s OddMuse [92], a single Perl script wiki install, and WikidPad [150], a single-user desktop-based wiki for notes and personal information management. Many are open source, free, and will often run on multiple operating systems. The differences between wikis are usually quite small but can include the development language used (Java, PHP, Python, Perl, Ruby, etc.), the database required (MySQL, flat files, etc.), whether attachment file uploading is allowed or not, spam prevention mechanisms, page access controls, RSS feeds, etc.

A typical wiki page has two specific buttons of interest: “Edit” and “History.” Normally, anyone can edit an existing wiki article, and if the article does not exist on a particular topic, anyone can create it. If someone messes up an article (either deliberately

or erroneously), there is a revision history – as in most wiki engines – so that the contents can be reverted or fixed by the community. Thus, while there is no predefined hierarchy in most wikis, content is auto-regulated thanks to an emergent consensus within the community, usually achieved in a delicate mix of democracy and meritocracy (e.g., most wikis include discussions pages where people can discuss sensible topics).

Going further than what was discussed previously, in semantic blogging it is not just blog posts that are being enhanced by structured metadata and semantics – this is happening in many other Social Web application areas. Wikis such as the Wikipedia have contained structured metadata in the form of templates for some time now, and at least 20 semantic wikis [118] have also appeared to address a growing need for more structure in wikis. In his presentation on “The Relationship Between Web 2.0 and the Semantic Web” [51], Mark Greaves said that semantic wikis are a promising answer to various issues associated with semantic authoring, by reducing the investment of time required for training on an annotation tool and by providing incentives required to providing semantic markup (attribution, visibility, and reuse by others).

Typical wikis usually enable the description of resources in natural language. By additionally allowing the expression of knowledge in a structured way, wikis can provide advantages in querying, managing, and reusing information. Wikis such as the Wikipedia have contained structured metadata in the form of templates for some time now (to provide a consistent look to the content placed within article texts), but there is still a growing need for more structure in wikis (e.g., the Wikipedia page about Ross Mayfield links to about 25 pages, but it is not possible to answer a simple question such as “find me all the organizations that Ross has worked with or for”). Templates can also be used to provide a structure for entering data, so that it is easy to extract metadata about the topic of an article (e.g., from a template field called “population” in an article about London). Semantic wikis bring this to the next level by allowing users to create semantic annotations anywhere within a wiki article’s text for the purposes of structured access and finer-grained searches, inline querying, and external information reuse. Generally, those annotations are designed to create instances of domain ontologies and their related properties (either explicit ontologies or ontologies that will emerge from the usage of the wiki itself), whereas other wikis use semantic annotations to provide advanced metadata regarding wiki pages. Obviously, both layers of annotation can be combined to provide advanced representation capabilities. By allowing people to add such extra metadata, the system can then show related pages (either through common relationships or properties, or by embedding search queries in pages). These enhancements are powered by the metadata that the people enter (aided by semantic wiki engines).

A semantic wiki should have an underlying model of the knowledge described in its pages, allowing one to capture or identify further information about the pages (metadata) and their relations. The knowledge model should be available in a formal language as RDFS or OWL, so that machines can (at least partially) process and reason on it. For example, a semantic wiki would be able to capture that an “apple” is a “fruit” (through an inheritance relationship) and present you with further fruits when you look at the apple article. Articles will have a combination of semantic data about the page itself

(the structure) and the object it is talking about (the content). Some semantic wikis also provide what is called inline querying. For example, in SemperWiki [119], questions such as “?page dc:creator EyalOren” (find me all pages where the creator is Eyal Oren) or “?s dc:subject todo” (show all me all my to do items) are processed as a query when the page is viewed and the results are shown in the wiki page itself [98]. Also, when defining some relationships and attributes for a particular article (e.g., “foaf:gender male”), other articles with matching properties can be displayed along with the article.

Moreover, some wikis such as IkeWiki [114] feature reasoning capabilities, for example, retrieving all instances of foaf:Person when querying for a list of all foaf:Agent(s) since the first class subsumes the second one in the FOAF ontology. Finally, just as in the semantic blogging scenario, wikis can enable the Web to be used as a clipboard, by allowing readers to drag structured information from wiki pages into other applications (e.g., geographic data about locations on a wiki page could be used to annotate information on an event or a person in your calendar application or address book software, respectively).

Semantic MediaWiki: The most widely used semantic wiki is Semantic MediaWiki [68], an extension to the popular MediaWiki system as used on the Wikipedia. Semantic MediaWiki allows for the expression of semantic data describing the connection from one page to another, and attributes or data relating to a particular page. Semantic MediaWiki is completely open in terms of the terms used for annotating content, such that the underlying data model, that is, the different ontologies used to model the instances, evolve according to user behavior.

Let us take an example of providing structured access to information via a semantic wiki. There is a Wikipedia page about JK Rowling that has a link to “Harry Potter and the Deathly Hallows” (and to other books that she has written), to Edinburgh because she lives there, and to Scholastic Press, her publisher. In a traditional wiki, you cannot perform fine-grained searches on the Wikipedia data set such as “show me all the books written by JK Rowling,” or “show me all authors that live in the UK,” or “what authors are signed to Scholastic,” because the type of links (i.e., the relationship type) between wiki pages are not defined. In Semantic MediaWiki, you can do this by linking with [[author of::Harry Potter and the Deathly Hallows]] rather than just the name of the novel. There may also be some attribute such as [[birthdate::1965-07-31]], which is defined in the JK Rowling article.

Such attributes can be used for answering questions like “show me authors over the age of 40” or for sorting articles, since this wiki syntax is translated into RDF annotations when saving the wiki page. Moreover, page categories are used to model the related class for the created instance. Indeed, in this tool, as in most semantic wikis that aim to model ontology instances, not only do the annotations make the link types between pages explicit, but they also make explicit the relationships between the concepts referred to in these wiki pages, thus bridging the gap from documents plus hyperlinks to concepts plus relationships. For instance, in the previous example, the annotation will not model that “the page about JK Rowling is the author of the page about Harry Potter and the Deathly Hallows” but rather that “the person JK Rowling is the author of the novel Harry Potter and the Deathly Hallows, and the pages describe these entities.”

The data within the wiki are exported using RDF and other export formats, so that the knowledge within the wiki can be reused in external applications. An example of such a reuse is the Beers of the World website [10], providing a sophisticated beer selector based on Exhibit [80] and drawing the data from an external Semantic MediaWiki installation.

Semantic MediaWiki is supported by a vibrant developer community, creating further extensions on top of the core system, and used in several hundred sites. The extensions add the ability for form-based editing, graphical querying, keyword-based structural queries, integration with mapping and geocoding services, exhibit visualization, and many others. The system has been localized to more than 40 languages.

OntoWiki: OntoWiki [94] is a semantic wiki developed by the AKSW research group at the University of Leipzig that also acts as an agile ontology editor and distributed knowledge engineering application. Unlike other semantic wikis, OntoWiki relies more on form-based mechanisms for the input of structured data rather than using syntax-based or markup-based inputs. One of the advantages of such an approach is that complicated syntaxes for representing structured knowledge can be hidden from wiki users and, therefore, syntax errors can be avoided. OntoWiki visually presents a knowledge base as an information map, with different views on available instance data. It aims to enable intuitive authoring of semantic content, and also features an inline editing mode for editing RDF content, similar to WYSIWIG for text documents. As with most wikis, it fosters social collaboration aspects by keeping track of changes and allowing users to discuss any part of the knowledge base, but OntoWiki also enables users to rate and measure the popularity of content, thereby honoring the activities of users. OntoWiki enhances the browsing and retrieval experience by offering semantically enhanced search mechanisms. Such techniques can decrease the entrance barrier for domain experts and project members to collaborate using semantic technologies. OntoWiki is open source and is based on PHP and MySQL.

SweetWiki: SweetWiki (Semantic Web Enabled Technology Wiki) [136] is a semantic wiki prototype featuring augmented-tagging features for end users. In contrast to the other wikis mentioned above, it is not designed for creating and maintaining ontology instances, but rather uses Semantic Web technologies to augment the user experience and navigation between pages. One relevant feature of SweetWiki regarding the work described in this chapter is the ability to organize tags as a hierarchy of concepts. This hierarchy is then modeled in RDFS so that it can be reused in other applications, while the wiki model itself is defined using a particular OWL ontology. Most importantly, this hierarchy of tags is not a personal one but is built and shared amongst all the users of the wiki. In this way, SweetWiki provides a social and collaborative approach to maintaining hierarchies of concepts that can be seen as lightweight ontologies. Moreover, users can define two tags as synonyms in order to solve heterogeneity issues. From a tagging point of view, tags can be not only assigned to Web pages but also to pictures and embedded videos, and these are then used to retrieve or browse content, while similar and related tags are used to augment the navigation process by suggesting related pages. Finally, SweetWiki models all of its data using RDFa. Hence, an application that wants to reuse it is only required to extract and parse an XHTML page, since all the required RDF annotations are embedded in it and can be extracted using GRDDL.

DBpedia: While not a semantic wiki per se, DBpedia benefits from wiki principles to build a large machine-readable knowledge base of structured and interlinked data [8]. It provides an RDF export of the Wikipedia and can be seen as one of the core components of the Linking Open Data [12] project (see [▶ Semantic Annotation and Retrieval: Web of Data](#)). Notably, various datasets in the Linking Open Data cloud (see [▶ Fig. 12.7 \[72\]](#)) link to DBpedia, since it is considered as being a central point [13] in efforts toward linking structured data on the Semantic Web.

DBpedia is created by exporting the “infoboxes” (i.e., metadata entered on Wikipedia articles using predefined template structures) from various language versions of Wikipedia and linking them together. By weaving Wikipedia articles and related objects into the Semantic Web, DBpedia defines URIs for many concepts so that people can use them in their semantic annotations. These various URIs can be used for instance in semantic tagging and semantic social bookmarking applications, as proposed in the MOAT framework (see [▶ Sect. 12.1.4](#)) or in Faviki [35] (see [▶ Sect. 12.2.4](#)). Another use case, especially related to social networking, is the reuse of these URIs to indicate interests of people, in combination with the foaf:topic interest property from the FOAF vocabulary. For example, to indicate that one is interested in Semantic Web technologies and the Social Web, he or she can use the following snippet of code. In this way, such information can be retrieved using SPARQL (see [▶ Querying the Semantic Web: SPARQL](#)), providing better ways to find people who are sharing a common interest in an online community.

```
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
@prefix dbpedia: <http://dbpedia.org/resource/>.
@prefix : <http://example.org/>.

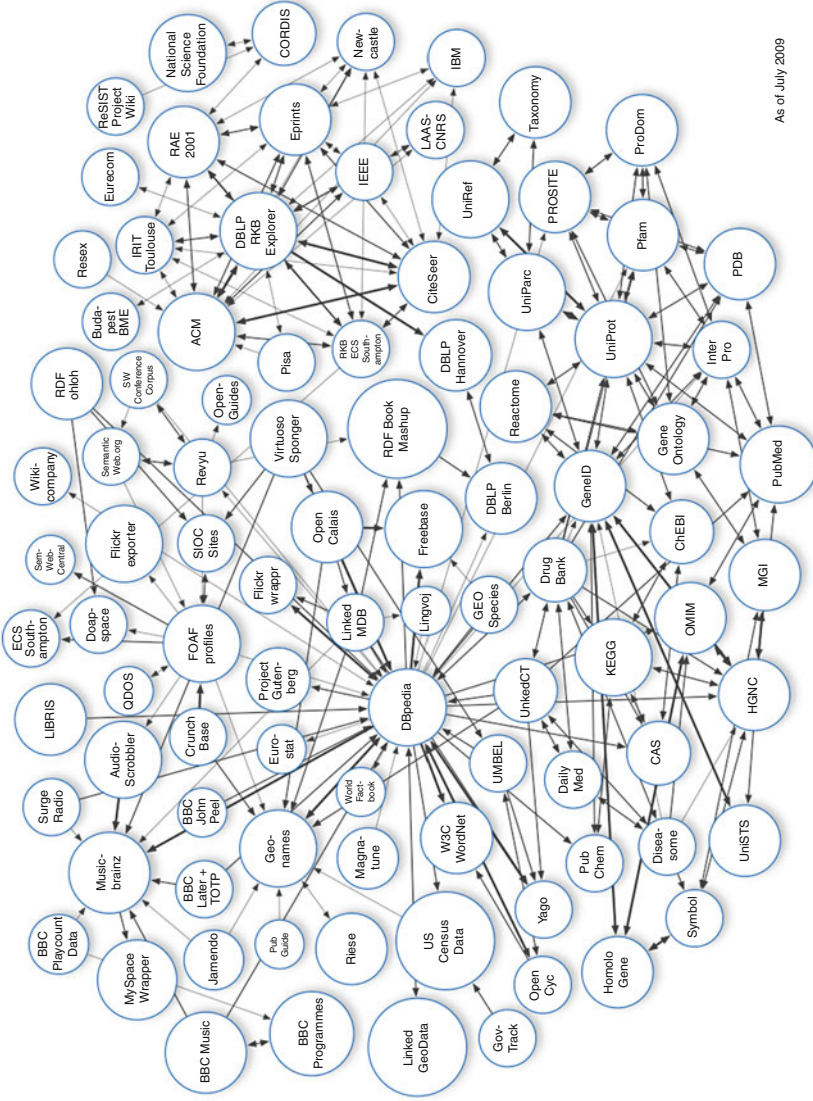
:me foaf:topic_interest dbpedia:Semantic_Web, dbpedia:Social_web.
```

In addition, the DBpedia dataset is freely available for download and it also provides a public SPARQL endpoint so that anyone can interact with it for advanced querying capabilities [26]. Other ways to navigate DBpedia include a faceted browser [23], where people can restrict content by type (using predefined facets as well as an auto-completion system), for example, “Scientist,” and then refine their queries based on various criteria, for example, birthdate [24].

An interesting application related to DBpedia is DBpedia Mobile [27], a “location-centric DBpedia client application for mobile devices” that consists of a map interface, the Marbles Linked Data Browser [74], and a GPS-enabled launcher application. The application displays nearby DBpedia resources (from a set of 300,000) based on a users’ geolocation as recorded through his or her mobile device. Efforts are also ongoing toward allowing DBpedia to feed new content back into the Wikipedia [67] (e.g., by suggesting new values for infoboxes, or by contributing back new maps created via DBpedia Mobile).

12.2.4 Semantic Social Bookmarking

Twine: Radar Networks is one of a number of startup companies that is applying Semantic Web technologies to social software applications. Radar’s flagship product



As of July 2009

Fig. 12.7

The Linking Open Data Cloud. CC by Richard Cyganiak and Anja Jentzsch – <http://lod-cloud.net>

is called Twine [144], and the company is led by CEO Nova Spivack. The Twine service allows people to share what they know and can be thought of as a knowledge networking application that allows users to share, organize, and find information with people they trust. People create and join “twines” (community containers) around certain topics of interest, and items (documents, bookmarks, media files, etc., that can be commented on) are posted to these twines through a variety of methods. Twine has a number of novel and useful functions that elevate it beyond the social bookmarking sites to which it has been compared, including an extensive choice of twineable item types, twined item customization (“add detail” allows user-chosen metadata fields to be attached to an item) and the “e-mail to a twine” feature (enabling twines to be populated through messages sent to a custom e-mail address).

The focus of Twine is these interests. Where Facebook [33] is often used for managing one’s social relationships and LinkedIn [70] is used for connections that are related to one’s career, Twine can be used for organizing one’s interests. With Twine, one can share knowledge, track interests with feeds, carry out information management in groups or communities, build or participate in communities around one’s interests, and collaborate with others. Twine allows people to find things that might be of interest to them based on what they are doing.

Twine performs natural language processing on text, mainly providing automatic tagging with semantic capabilities. It has an underlying ontology with a million instances of thousands of concepts to generate these tags (at present, Twine is exposing just some of these). Radar Networks is also working on statistical analysis and machine-learning approaches for clustering related content to show people, items and interests that are related to each other (e.g., to give information to users such as “here is a selection of things that are all about movies you like”).

Everything in Twine is generated from an ontology. Even the site itself – user interface elements, sidebars, navigation bar, buttons, etc. – come from an application-definition ontology. Similarly, the Twine data is modeled on a custom ontology. However, Twine is not just limited to these internal ontologies, and Radar Networks is beginning the process of bringing in other external ontologies and using them within Twine. They offer a lightweight ontology editing tool to allow people to make their own ontologies (e.g., to express domain-specific content) resulting in the Twine community having a more extensible infrastructure.

Twine search also has semantic capabilities. For example, bookmarks can be filtered by the companies they are related to, or people can be filtered by the places they are from. Underneath Twine, a lot of research work on scaling has been carried out, but it is not trying to index the entire Web. However, Twine does pull in related objects (e.g., from links in an e-mail), thereby capturing information around the information that you bring in and that you think is important. In terms of data interoperability, semantic data can be obtained from Twine in RDF for reuse elsewhere (by appending “?rdf” to the end of any Twine URL). Having already hardcoded some interoperability with services like Amazon [3] and provided import functionality from Delicious [28], Radar Networks is also looking at potential adaptors to other services including Digg [29], desktop bookmark files, Outlook contacts, Lotus Notes, Exchange, and Freebase [43].

Twine is aiming at mainstream users, so the interface has to be simple so that someone who knows nothing about structured data or automatic tagging should be able to figure out in a few minutes or even seconds how to use it. Individuals are Twine's first target market, allowing them to author and develop rich semantic content. For example, this could be a professional who has a need for a particular interest in some technical subject that is outside the scope of what they are doing at the moment. However, such a service becomes more valuable when users are connected to other people, if they join groups, thereby giving a richer network effect. The main value proposition for these users is that they can keep track of things they like, people they know, and capturing knowledge that they think is important.

When groups start using Twine, collective intelligence begins to take place (by leveraging other people who are researching material, finding items, testing, commenting, etc.). It is a type of communal knowledge base similar to other services like Wikipedia or Freebase. However, unlike many public communal sites, in Twine more than half of the data and activities are private (60%). Therefore privacy and permission control is very important, and it is deeply integrated into the Twine data structures. Since Twine left beta in 2008, public twines have become visible to search engines and SEO has been applied to increase the visibility of this content.

With such a service, there is a requirement for duplication detection. Most people submit similar bookmarks and it is reasonably straightforward to identify these, for example, when the same item enters the system through different paths and has different URLs. However, some advanced techniques are required when the content is similar but comes from different locations on the Web.

Referring to the theory of object-centered sociality [66] and others (i.e., people are networked through shared objects of interest), there is great potential in the community aspects of twines. These twines can act as “social objects” that will draw people back to the service in a far stronger manner than other social bookmarking sites currently do (in part, this is due to there being a more identifiable home for these objects and also due to the improved commenting facilities that Twine provides).

Faviki: Faviki [35] is a social bookmarking service that uses a controlled vocabulary for its tags, namely, the resources defined in Wikipedia. Hence, it provides features such as multilingual tagging (with various tags being automatically linked to the same concept), a related tags suggestion service (based on the relationships between these concepts), and it can also display tag descriptions. Faviki relies on DBpedia [25], Zemanta [159], and Google Language APIs [48] to provide its service. In addition, Faviki exposes its data in RDFa using the Common Tag format.

12.2.5 Review Websites

Revyu.com: Revyu.com [56, 110] is an online service dedicated to creating reviews for various items ranging from conference papers to pubs or restaurants (► Fig. 12.8). It reuses some well-known principles and features of Social Web applications such as tags,

The screenshot shows the Revyu.com website interface. At the top, there are navigation links: Home | Browse Things | Search Things | Browse People, and Login/Register | New Review. The main content area is titled "Things Tagged ireland (6)". Below this, there is a list of items with their respective tags:

- # "Found Out" Cafe, Inishannon, County Cork, Ireland
- I Tinnacullen (Garinish Island), Glengarriff, County Cork, Ireland
- Imagine Ireland - Holiday Accommodation
- Imperial Hotel, Galway, Ireland
- L La Jolie Brise, Baltimore, County Cork, Ireland
- T The Gallery, Ballymacrown Homestead, Baltimore, County Cork, Ireland

At the bottom of the page, there are links for Revyu.com: Contact | Credits | Privacy Policy | Disclaimer. An overlaid window on the right side of the page shows the RDF metadata for the tag 'ireland'. The metadata is displayed as follows:

```

<rdf:RDF xml:base="http://revyu.com/">
  <tag:Tag rdf:about="tags/ireland">
    <rdf:label>ireland</rdf:label>
  </tag:Tag>
  <owl:Thing rdf:about="things/-found-out-cafe-inishannon-county-cork-ireland">
    <tag:tag rdf:resource="taggings/df70086c1fcf51cab468e2fd03ac2e6452c5f10"/>
  </owl:Thing>
  <tag:Tagging rdf:about="taggings/df70086c1fcf51cab468e2fd03ac2e6452c5f10">
    <tag:associatedTag rdf:resource="tags/ireland"/>
  </tag:Tagging>
  <owl:Thing rdf:about="things/la-jolie-brise-baltimore-county-cork-ireland">
    <tag:tag rdf:resource="taggings/d2186e3e02757bf7ffe8c6a10c1b38211d97766"/>
  </owl:Thing>
  <tag:Tagging rdf:about="taggings/d2186e3e02757bf7ffe8c6a10c1b38211d97766">
    <tag:associatedTag rdf:resource="tags/ireland"/>
  </tag:Tagging>
  <owl:Thing rdf:about="things/imagine-ireland-holiday-accommodation">
    <tag:tag rdf:resource="taggings/5c90e05c3db3569824a3fd3b9233bf6321c40333"/>
  </owl:Thing>
  <tag:Tagging rdf:about="taggings/5c90e05c3db3569824a3fd3b9233bf6321c40333">
    <tag:associatedTag rdf:resource="tags/ireland"/>
  </tag:Tagging>

```

Fig. 12.8

The Revyu service

tag clouds, and stars ratings, and it provides a JavaScript bookmarklet to ease the publication of new reviews for end users when browsing the Web. Most importantly, Revyu.com is completely RDF based.

Each review is modeled using the RDF Review vocabulary [109] (compatible with the hReview microformat), and tags as well as tagging actions are represented using the Tag Ontology.

As Revyu.com also provides a SPARQL endpoint for querying its data, it also allows one to reuse tagged data from the website in any other application, as well as enabling mash-ups with existing content. Two important features of Revyu.com regarding the use of Semantic Web technologies are:

- *Integration and Interlinking with Other Data Sets:* Thanks to different heuristics, Revyu.com integrates identity links (using owl:sameAs properties) to resources already defined on the Semantic Web, especially resources being described in datasets from the Linking Open Data cloud (see [Semantic Annotation and Retrieval: Web of Data](#)). For example, most reviews regarding research papers are linked to the paper definition from the Semantic Web Dogfood project [30], while reviews about movies can be automatically linked to their DBpedia URI. Thus, it provides global interlinking of Semantic Web resources rather than defining new URIs for existing concepts.
- *The ability to consume FOAF-based user profiles:* While many Social Web applications require the user to fill in their personal details when subscribing, with those details having already been filled in on other platforms, Revyu.com allows one to simply give

his or her FOAF URI so that the information contained therein is automatically reused. Consuming FOAF profiles in Web-based applications provides a first step toward solving data portability issues between applications on the Social Web.

Therefore, Revyu.com combines Web 2.0-type interfaces and principles such as tagging with Semantic Web modeling principles to provide a reviews website that follows the principles of the Linking Open Data initiative. Anyone can review objects defined on other services (such as a movie from DBpedia), and the whole content of the website is available in RDF, therefore it is available for reuse by other Social Semantic Web applications.

12.2.6 Social Semantic Web Applications for Sharing Scientific Research

Science Collaboration Framework: The Science Collaboration Framework (SCF) is a reusable, open-source platform for structured online collaboration in biomedical research that leverages existing biomedical ontologies and RDF resources on the Semantic Web. The SCF GPL software [113] consists of the Drupal core content management system and customized modules. SCF supports structured Social Web-type community discourse amongst biomedical research scientists that is centered on a variety of interlinked heterogeneous data resources available to them (both formal and informal content, including scientific articles, news items, interviews, and various other perspectives).

The first instance of the SCF framework is being used to create an open-access online community for stem cell research called StemBook [130]. StemBook was developed based on requirements from the Harvard Stem Cell Institute. A second community is being planned for PD Online, a Web community for Parkinson's disease researchers sponsored by the Michael J. Fox Foundation (MJFF). The developers of SCF have cited significant overlaps between PD Online requirements and existing features built for StemBook, suggesting that the framework will achieve feature convergence through successive community implementations.

The architecture [22] makes it possible to define common schemas in OWL for a set of Web communities and to enable interoperability across biological resources, SWAN research statements or other objects of interest defined in the shared schemas. It is planned to make these graphs available via RDFa embedded within the HTML, and this work is being carried out in parallel with efforts to integrate RDFa into Drupal core [20]. myExperiment [87] is a collaborative environment where scientists can publish their experimental results and the workflows they used to produce these results. The myExperiment team reuses existing vocabularies for publishing and sharing experimental data by scientists. In David Newman's myExperiment ontology [86], concepts from Dublin Core, FOAF, and SIOC are reused since they are closely related to the two main functions of myExperiment: a social networking framework for researchers, and a metadata registry for experiments.

12.3 Future Issues

12.3.1 Searching the Social Semantic Web

As has been described already, RDF can be used to structure and expose information from the Social Web allowing the simple generation of semantic mash-ups and integrated views for both proprietary and public information. HTML content can also be made compatible with RDF through RDFa (RDF annotations embedded in XHTML attributes), thereby enabling effective semantic search without requiring one to crawl a new set of pages (e.g., the Common Tag effort allows metadata and URIs for tags to be exposed using RDFa and shared with other applications).

Search engine companies have recently started to publish recommended ontologies so that site owners and publishers can use these to annotate their content with RDFa or microformats. The result is that more relevant content will be displayed in search results, and the results themselves become more visually appealing (by using metadata to show people or organizations on a map, to display stars for review ratings, etc.), thereby encouraging click throughs.

Yahoo! SearchMonkey [116] has published a list of recommended vocabularies (including FOAF, GoodRelations, hReview, SIOC, vCalendar, and vCard) that publishers can use to create structured data and thereby drive more traffic to their sites. Google's "Rich Snippets" initiative (introduced in May 2009 [49]) has a similar aim, albeit using Google's own RDF vocabularies rather than popular existing ones like FOAF. Rich Snippets also promotes the use of the hReview, hProduct, and hCard microformats for annotating reviews, products, and people or organizations, respectively.

By providing RDFa-enabled HTML templates for popular social software applications with metadata relevant for search results (leveraging experience gained from creating RDF/XML exporters for WordPress, vBulletin, etc.), a very important step toward the formation of the Social Semantic Web can be taken. The RDFa in Drupal initiative is one of the first efforts to do this.

The SPARQL query language can be used for searching not just for keywords but for relationships between people and objects in aggregated Social Semantic Web data. Using RDF and SPARQL, it becomes possible to integrate diverse information from heterogeneous social websites, enabling improved navigation and the ability to query over these data. There are also advantages for those interested in studying social networks or looking for less obvious connections between people, as the Semantic Web makes large-scale, multi-relational datasets freely available for analysis. While the majority of SPARQL interfaces are designed for use within application architectures, more specialized user-oriented interfaces to custom SPARQL queries could provide network visualizations based on implicit and explicit relationships. There may also be new business models based on SPARQL queries across these data aggregates, for example, to provide topic-centric advertising on one site based on the related linked objects (and associated topics) from other sites.

A sample query for extracting the social network formed by explicit foaf:knows relationships follows using the SPARQL query language:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT DISTINCT ?s ?o
WHERE {
  ?s a foaf:Person.
  ?o a foaf:Person.
  ?s foaf:knows ?o.
}
```

In addition to explicitly stated person-to-person links, there are many implicit social connections present on the Web. A sample query for extracting the implicit social network formed by replies to posts follows (using the has_reply property from SIOC).

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX sioc: <http://rdfs.org/sioc/ns#>
SELECT ?author1 ?author2
WHERE {
  ?post1 a sioc:Post;
    foaf:maker ?author1;
    sioc:has_reply ?post2.
  ?post2 a sioc:Post;
    foaf:maker ?author2.
}
```

12.3.2 Trust and Privacy on the Social Semantic Web

Some challenges must also be overcome regarding the online identity aspect of the Social Web, as well as authentication and privacy for users of social websites. An interesting aspect of social networking and social websites is that most people use various websites because they want to fragment their online identity: uploading pictures of friends on MySpace, forming business contacts on LinkedIn, etc. Under each persona, a user may reveal completely different facets of their personality. People may wish to share many of their identities with certain contacts, but retain more privacy when dealing with others. For example, many people are careful to keep their personal life distinct from their professional life. However, just as people may wish to keep separate identities for some purposes, it can also be beneficial to be able to connect these persons as and when desired. Members of online communities often expend a lot of effort into forming relationships and building their reputation. Since reputation determines how much trust other people will place in an individual, it can be of very real value and therefore the ability to maintain a reputation across different identities could be very beneficial.

While the Semantic Web and, in particular, reasoning principles (such as leveraging Inverse Functional Properties (IFPs)) allow one to merge these data and provide vocabularies, methods, and tools for data portability among social websites, this intentional

identity fragmentation must be taken into account on the Social Semantic Web. It implies a need for new ways to authenticate queries or carry out inferencing, by delivering data in different forms depending on, for instance, which social subgraph the person requesting the data belongs to (family, coworker, etc.). Here, Web 2.0 efforts like OAuth [91] are of interest. OAuth is an open protocol that enables users to grant applications access to their protected data stored in accounts they hold with other services. Also relevant is the recent proposal for FOAF+SSL [131, 132]. Moreover, advanced social aspects of contextualizing information delivery may be added later. The nature of each relationship (e.g., work, family, romantic, friendship) could be taken into account, as well as the current status, location, or even the mood of a user. In some cases, external influences such as the political climate in a country may be considered in determining what kind of information to share about an individual. Additionally, as relationships evolve over time, the processing of requests could be updated accordingly.

Besides the issue of security, the issue of privacy also arises due to the possibility to more easily combine existing data sources. In the USA, donors for political campaigns, be it for a candidate or a proposition, are listed publicly. This improves transparency and aims at countering hidden influences in policy-making and the democratic process. In November 2008 in California, voters passed proposition 8, which overturned the right of same-sex couples to marry. The campaigns concerning the propositions raised a record of over 80 million US Dollars – and as always, the list of donors was released to the public. The list contained names and addresses of the donors. The addresses were geo-coded and a mash-up with Google Maps was created to provide a map of all donors for proposition 8 (see screenshot in [Fig. 12.9](#) [32]). This is just one example how data can be merged. In the future, we expect the Semantic Web technology to simplify the creation of such mash-ups even further, so that even nontechnical users will be able to create such views on data on-the-fly. The impact of these technologies on privacy is yet only marginally understood, and protocols like P3P [102] (the Platform for Privacy Preferences) as well as specifications like CC REL [18] (for including licenses in content) are important here.

12.3.3 Integrating with the Social Semantic Desktop

The vision of the Social Semantic Web has emerged in parallel with another prominent research area, that of the Social Semantic Desktop [117], where users and their peers can share and interlink multimedia content, calendars, e-mails, or documents. There is a convergence occurring between these two areas, for example, in the areas of information models (for personal or shared content), collaboration and knowledge exchange, domain-specific structures, expert finding, argumentative discussions, and semantic authoring and publishing. The combination of these two areas is leading to what has been termed Social Semantic Information Spaces [62].

In [Sect. 12.2.1](#) an example of an overlap between the Social Semantic Web and the Social Semantic Desktop was given, where a blogging application allowed one to leverage

PROP 8 MAPS

A mash-up of [Google Maps](#) and [Prop 8 Donors](#).

Proposition 8 changed the California state constitution to prohibit same-sex marriage. These are the people who donated in ord

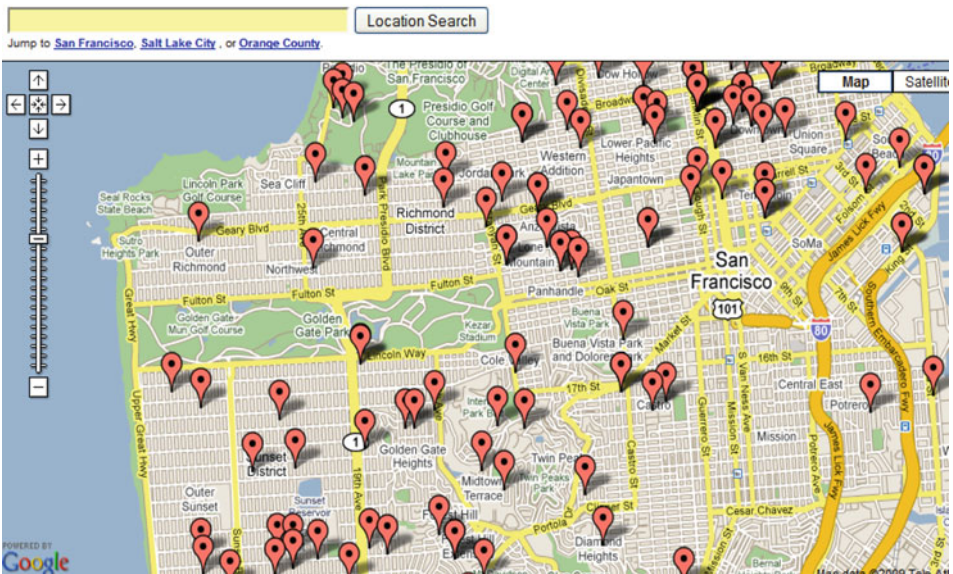


Fig. 12.9

A screenshot of EightMaps, a mash-up of Proposition 8 donors and Google Maps [32]

information from the desktop (e.g., in an address book or event calendar) and lift the associated metadata onto the Web for reuse by others. Similar integrations can occur with desktop-based microblogging solutions or mobile phone photo applications. One particular project of interest and referenced earlier is the Nepomuk social semantic desktop [90], a European Union-funded project that focused on how semantic technologies can help people to find and add structure to information on their personal computers, and to share that information with other users. Nepomuk allows users to give meaning to documents, contact details, pictures, videos, and other data files stored on a user's computer, regardless of the file format, associated application, or language used, making it easier and quicker to find information and to identify connections between different items. When information is added to the desktop, Nepomuk asks users to annotate the information so that it can be correctly situated, and it also crawls the user's computer to search for existing information thereby establishing connections between different content items. Nepomuk is available in KDE4, a popular desktop environment for Linux.

12.4 Cross-References

- Knowledge Management in Large Organizations
- Ontologies and the Semantic Web

- ▶ Querying the Semantic Web: SPARQL
- ▶ Semantic Annotation and Retrieval: RDF
- ▶ Semantic Annotation and Retrieval: Web of Data

Acknowledgments

The work presented in this chapter was supported by the Lion-2 project supported by Science Foundation Ireland under Grant No. SFI/08/CE/I1380.

Supported by the EU IST project ACTIVE, <http://www.active-project.eu>.

References

1. Abel, E.: The benefit of additional semantics in folksonomy systems. In: Proceeding of the Second Ph.D. Workshop on Information and Knowledge Management (PIKM 2008), Napa Valley, pp. 49–56. ACM Press, New York (2008)
2. <http://www.google.com/adsense>
3. <http://www.amazon.com>
4. Ankolekar, A., Vrandečić, D., Kröttsch, M., Tran, D.T.: The two cultures: mashing up web 2.0 and the semantic web. In: Proceedings of the 16th International Conference on the World Wide Web (WWW 2007), Banff. ACM Press, New York (2007)
5. <http://apml.areyoupayingattention.com>
6. <http://arc.semsol.org>
7. <http://www.techcrunch.com/2008/05/05/twitter-can-be-liberated-heres-how>
8. Auer, S., Bizer, C., Lehmann, J., Kobilarov, G., Cyganiak, R., Ives, Z.: Dbpedia: a nucleus for a web of open data. In: Proceedings of the Sixth International Semantic Web Conference and Second Asian Semantic Web Conference (ISWC/ASWC 2007), Busan. Lecture Notes in Computer Science, vol. 4825, pp. 722–735. Springer, Berlin
9. http://www.bbc.co.uk/blogs/technology/2008/05/twitter_and_the_china_earthqua.html
10. <http://beer.geekworks.de>
11. <http://www.bibsonomy.org>
12. Bizer, C., Heath, T., Berners-Lee, T.: Linked data – the story so far. *Int. J. Semant. Web Inf. Syst.* 5, 1–22 (2009)
13. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia – a crystallization point for the web of data. *J. Web Semant.* 7, 154–165 (2009)
14. Breslin, J.G., Harth, A., Bojars, U., Decker, S.: Towards semantically-interlinked online communities. In: Proceedings of the Second European Semantic Web Conference (ESWC 2005), Heraklion. Lecture Notes in Computer Science, vol. 3532, pp. 500–514. Springer, Heidelberg (2005)
15. <http://danbri.org/words/2007/11/04/223>
16. Brickley, D., Miller, L.: FOAF vocabulary specification. Namespace document, FOAF project. <http://xmlns.com/foaf/0.1/> (2004)
17. Cayzer, S.: Semantic blogging and decentralized knowledge management. *Commun. ACM* 47(12), 47–52 (2004)
18. http://wiki.creativecommons.org/CC_REL
19. <http://www.commontag.org>
20. Corlosquet, S., Delbru, R., Clark, T., Polleres, A., Decker, S.: Produce and consume linked data with drupal! In: Proceedings of the Eighth International Semantic Web Conference (ISWC 2009), Chantilly. Lecture Notes in Computer Science, vol. 5823, pp. 763–778. Springer, Berlin (2009)
21. <http://www.craigslist.org>
22. Das, S., Greena, T., Weitzman, L., Lewis-Bowen, A., Clark, T.: Linked data in a scientific collaboration framework. In: Proceedings of the 17th International Conference on the World Wide Web (WWW 2008), Beijing (2008)
23. <http://dbpedia.neofonie.de/browse/>
24. <http://dbpedia.neofonie.de/browse/rdf-type:Scientist/>
25. <http://dbpedia.org>
26. <http://dbpedia.org/sparql/>
27. <http://wiki.dbpedia.org/DBpediaMobile>
28. <http://www.delicious.com>
29. <http://digg.com>

30. <http://data.semanticweb.org>
31. Dolog, P., Krötzsch, M., Schaffert, S., Vrandečić, D.: Social web and knowledge management. In: King, I., Baeza-Yates, R.A. (eds.) *Weaving Services and People on the World Wide Web*, pp. 217–227. Springer, Berlin (2008)
32. <http://www.eightmaps.com>
33. <http://www.facebook.com>
34. <http://www.dcs.shef.ac.uk/~mrowe/foafgenerator.html>
35. <http://www.faviki.com>
36. Fernández, S., Berrueta, D., Labra, J.E.: Mailing lists meet the semantic web. In: *Proceedings of the BIS 2007 Workshop on Social Aspects of the Web (SAW 2007)*, Poznan. CEUR, vol. 245. CEUR-WS.org (2007)
37. <http://www.flickr.com>
38. <http://apassant.net/blog/2007/12/18/rdf-export-flickr-profiles-foaf-and-sioc/>
39. <http://www.foaf-project.org>
40. <http://www.ldodds.com/foaf/foaf-a-matic>
41. <http://foafbuilder.qdos.com>
42. Franz, T., Staab, S.: SAM: semantics aware instant messaging for the networked semantic desktop. In: *Proceedings of the First Workshop on the Semantic Desktop (SemDesk 2005)*, Fourth International Semantic Web Conference (ISWC 2005), Galway. CEUR, vol. 175. CEUR-WS.org (2005)
43. <http://www.freebase.com>
44. <http://www.freedb.org>
45. <http://friendfeed.com>
46. <http://www.friendster.com>
47. <http://www.genomes.org>
48. <http://code.google.com/apis/ajaxlanguage/>
49. <http://www.google.com/support/webmasters/bin/answer.py?answer=99170>
50. <http://maps.google.com>
51. <http://rease.semanticweb.org/ubp/PUSH/search@srchDetailsLR?lrID=lr-lear-diederich-1186478671106>
52. <http://gregarius.net>
53. Gruber, T.: Ontology of folksonomy: a mash-up of apples and oranges. *Int. J. Semant. Web Inf. Syst.* **3**(2), 1–11 (2007)
54. Harth, A., Gassert, H., O’Murchu, I., Breslin, J.G., Decker, S.: WikiOnt: an ontology for describing and exchanging wikipedia articles. In: *Proceedings of Wikimania 2005 – The First International Wikimedia Conference*, Frankfurt (2005)
55. <http://microformats.org/wiki/hcard>
56. Heath, T., Motta, E.: Revyu.com: a reviewing and rating site for the web of data. In: *Proceedings of the Sixth International Semantic Web Conference and Second Asian Semantic Web Conference (ISWC/ASWC 2007)*, Busan. *Lecture Notes in Computer Science*, vol. 4825, pp. 895–902. Springer, Heidelberg (2007)
57. Hendler, J.A., Golbeck, J.: Metcalfé’s law, web 2.0, and the semantic web. *J. Web Semant.* **6**(1), 14–20 (2008)
58. <http://hi5.com>
59. <http://www.housingmaps.com>
60. <http://hyperdata.org/xmlns/ibis/>
61. <http://identi.ca>
62. <http://www2006.org/tutorials/#T13>
63. <http://esw.w3.org/topic/IswcPodcast>
64. Karger, D.R., Quan, D.: What would it mean to blog on the semantic web? In: *The Semantic Web: Third International Semantic Web Conference (ISWC 2004)*, Hiroshima. *Lecture Notes in Computer Science*, vol. 3298, pp. 214–228. Springer, Heidelberg (2004)
65. Kim, H.L., Yang, S.-K., Breslin, J.G., Kim, H.-G.: Simple algorithms for representing tag frequencies in the SCOT exporter. In: *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, Silicon Valley, pp. 536–539. IEEE Computer Society, Washington, DC (2007)
66. Knorr-Cetina, K.: Sociality with objects: social relations in postsocial knowledge societies. *Theory Cult. Soc.* **14**(4), 1–30 (1997)
67. http://videlectures.net/www09_kobilarov_dbpldh/
68. Krötzsch, M., Vrandečić, D., Völkel, M.: Semantic MediaWiki. In: *Proceedings of the Fifth International Semantic Web Conference (ISWC 2006)*, Athens, GA. *Lecture Notes in Computer Science*, vol. 4273, pp. 935–942. Springer, Heidelberg (2006)
69. <http://last.fm>
70. <http://www.linkedin.com>
71. <http://www.livejournal.com>
72. <http://richard.cyganiak.de/2007/10/lod/>
73. <http://www.loudervoice.com>
74. <http://marbles.sourceforge.net>
75. McAfee, A.P.: Enterprise 2.0: the dawn of emergent collaboration. *MIT Sloan Manag. Rev.* **47**(3), 21–28 (2006)
76. <http://www.mediawiki.org>
77. <http://www.microsyntax.org>

78. <http://buzzword.org.uk/2009/microturtle/>
79. Mika, P.: Ontologies are us: a unified model of social networks and semantics. In: Proceedings of the Fourth International Semantic Web Conference (ISWC 2005), Sardinia. Lecture Notes in Computer Science, vol. 3729, pp. 522–536. Springer, Heidelberg (2005)
80. <http://www.simile-widgets.org/exhibit/>
81. <http://www.moat-project.org>
82. <http://www.moat-project.org/architecture>
83. Möller, K., Bojars, U., Breslin, J.G.: Using semantics to enhance the blogging experience. In: Proceedings of the Third European Semantic Web Conference (ESWC 2006), Budva. Lecture Notes in Computer Science, vol. 4011, pp. 679–696. Springer, Heidelberg (2006)
84. Möller, K., Decker, S.: Harvesting desktop data for semantic blogging. In: Decker, S., Park, J., Quan, D., Sauermann, L. (eds.) Proceedings of the Semantic Desktop (SD 2005), Workshop at the ISWC 2005, Galway (2005)
85. <http://www.mybloglog.com>
86. <http://users.ecs.soton.ac.uk/drn05r/release/stable/myexp.owl>
87. <http://www.myexperiment.org>
88. <http://www.myspace.com>
89. <http://network.nature.com>
90. <http://nepomuk.semanticdesktop.org>
91. <http://www.oauth.net>
92. <http://www.oddmuse.org>
93. <http://www.opengraphprotocol.org/>
94. <http://www.ontowiki.net>
95. <http://www.openid.net>
96. <http://www.milanstankovic.org/opo/>
97. O'Reilly, T.: O'Reilly network: what is web 2.0: design patterns and business models for the next generation of software. <http://www.oreillynet.com/lpt/a/6228> (2005)
98. Oren, E., Völkel, M., Breslin, J.G., Decker, S.: Semantic wikis for personal knowledge management. In: Proceedings of the 17th International Conference on Database and Expert Systems Applications (DEXA 2006), Krakow. Lecture Notes in Computer Science, vol. 4080, pp. 509–518. Springer, Heidelberg (2006)
99. <http://www.orkut.com>
100. Orlandi, F., Passant, A.: Enabling cross-wikis integration by extending the SIOC ontology. In: Proceedings of the Fourth Workshop on Semantic Wikis (SemWiki 2009), Heraklion (2009)
101. Osterfeld, F., Kiesel, M., Schwarz, S.: Nabu – a semantic archive for XMPP instant messaging. In: Proceedings of the First Workshop on the Semantic Desktop (SemDesk 2005), Fourth International Semantic Web Conference (ISWC 2005), Galway. CEUR, vol. 175. CEUR-WS.org (2005)
102. <http://www.w3.org/P3P/>
103. Passant, A.: me OWL:sameAs flickr:33669349@N00. In: Proceedings of the WWW 2008 Workshop Linked Data on the Web (LDOW 2008), Beijing. CEUR, vol. 369. CEUR-WS.org (2008)
104. Passant, A., Laublet, P., Breslin, J.G., Decker, S.: A URI is worth a thousand tags: from tagging to linked data with MOAT. *Int. J. Semant. Web Inf. Syst.* 5, 71–94 (2009)
105. <http://purplewiki.blueoxen.net>
106. Quan, D., Huynh, D., Karger, D.R.: Haystack: a platform for authoring end user semantic web applications. In: Proceedings of the Second International Semantic Web Conference (ISWC 2003), Sanibel Island. Lecture Notes in Computer Science, vol. 2870, pp. 738–753. Springer, Heidelberg (2003)
107. Rehatschek, H., Hausenblas, M.: Enhancing the exploration of mailing list archives through making semantics explicit. In: Proceedings of the Semantic Web Challenge 2007, Collocated with the Sixth International Semantic Web Conference (ISWC 2007), Seoul (2007)
108. <http://vocab.org/relationship/>
109. <http://www.purl.org/stuff/rev#>
110. <http://revyu.com>
111. Rowe, M., Ciravegna, F.: Getting to me – Exporting semantic social network from facebook. In: Proceedings of the First Workshop on Social Data on the Web (SDoW 2008), Karlsruhe. CEUR, vol. 405. CEUR-WS.org (2008)
112. <http://web.resource.org/rss/1.0/>
113. <http://sciencecollaboration.org>
114. Schaffert, S.: IkeWiki: a semantic wiki for collaborative knowledge management. In: Proceedings of the First International Workshop on Semantic Technologies in Collaborative Applications (STICA 2006), Manchester (2006)
115. <http://www.scot-project.org>
116. <http://developer.yahoo.com/searchmonkey/>
117. <http://www.semanticdesktop.org>
118. http://semanticweb.org/index.php/Semantic_Wiki_State_Of_The_Art
119. <http://www.eyaloren.org/semperwiki.html>

120. <http://kantenwerk.org/shift>
121. <http://www.sioc-project.org>
122. <http://www.w3.org/Submission/2007/02/>
123. <http://www.w3.org/2004/02/skos/>
124. <http://www.smesher.org>
125. <http://smob.sioc-project.org>
126. <http://www.socialtext.com>
127. <http://bZR.mfd-consult.dk/sparqlpress>
128. http://civilities.net/Star_Priority_Notation
129. <http://status.net>
130. <http://www.stembook.org>
131. http://blogs.sun.com/bblfish/entry/rdfauth_sketch_of_a_buzzword
132. Story, H., Harbulot, B., Jacobi, I., Jones, M.: FOAF+TLS: restful authentication for the social web. In: Proceedings of the First Workshop on Trust and Privacy on the Social and Semantic Web (SPOT 2009), Heraklion. CEUR Workshop Proceedings, vol. 447. CEUR-WS.org (2009)
133. <http://www.structuredblogging.org>
134. <http://salt.semanticauthoring.org>
135. <http://www.w3.org/TR/hcls-swansioc/>
136. <http://sweetwiki.inria.fr/ontology>
137. <http://www.tagora-project.eu/>
138. <http://www.holygoat.co.uk/projects/tags/>
139. Tapscott, D., Williams, A.D.: Wikinomics: How Mass Collaboration Changes Everything. Pearson Education, New York (2007)
140. <http://technorati.com/blogging/state-of-the-blogsphere/>
141. <http://technorati.com/weblog/2007/04/328.html>
142. <http://www.telegraph.co.uk/news/worldnews/asia/india/3530640/Mumbai-attacks-Twitter-and-Flickr-used-to-break-news-Bombay-India.html>
143. <http://www.tweetdeck.com>
144. <http://www.twine.com>
145. <http://twitter.com>
146. <http://semantictweet.com/>
147. <http://upcoming.org>
148. <http://www.ietf.org/rfc/rfc2426.txt>
149. Völkel, M., Oren, E.: Towards a Wiki Interchange Format (WIF) – opening semantic wiki content and metadata. In: Proceedings of the First Workshop on Semantic Wikis – From Wiki to Semantics (SemWiki 2006), Budva. CEUR, vol. 206. CEUR-WS.org (2006)
150. <http://wikidpad.sourceforge.net>
151. <http://en.wikipedia.org>
152. http://en.wikipedia.org/wiki/Metcalfes_Law
153. <http://wordpress.org>
154. <http://www.gmpg.org/xfn>
155. <http://vocab.sindice.com/xfn.html>
156. <http://answers.yahoo.com>
157. <http://www.yammer.com>
158. <http://www.youtube.com>
159. <http://www.zemanta.com>

13 Ontologies and the Semantic Web

Stephan Grimm¹ · Andreas Abecker^{1,4} · Johanna Völker² · Rudi Studer^{1,3}

¹FZI Research Center for Information Technology, Karlsruhe, Germany

²University of Mannheim, Mannheim, Germany

³Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

⁴disy Informationssysteme GmbH, Karlsruhe, Germany

13.1	<i>Scientific and Technical Overview: Foundations</i>	509
13.1.1	Notion of Ontologies	509
13.1.1.1	Origin and Definition	510
13.1.1.2	Essential Characteristics of an Ontology	511
13.1.1.3	Formal Ontology Model	513
13.1.2	Ontologies in Information Systems	515
13.1.2.1	Ontologies and Formal Knowledge Representation	515
13.1.2.2	Usage of Ontologies	517
13.1.2.3	Ontology Languages	519
13.1.3	Variants of Ontologies	521
13.1.3.1	Varying Form of Appearance	521
13.1.3.2	Varying Scope	522
13.1.3.3	Varying Degree of Formality	523
13.1.3.4	Some Example Semantic Web Ontologies	525
13.2	<i>Scientific and Technical Overview: Engineering and Methodological Aspects</i>	527
13.2.1	Manual Ontology Construction	528
13.2.1.1	Ontology Editors	528
13.2.2	Methodologies for Ontology Engineering	529
13.2.2.1	Generic Methodology	530
13.2.2.2	Methodologies and Tool Support	531
13.2.3	Ontology Learning	532
13.2.3.1	Methods for Ontology Learning from Text	533
13.2.3.2	Tools and Implementations	535
13.2.4	Methodological Aspects of Ontology Learning	537

13.3	<i>Example Applications</i>	538
13.3.1	Generic Functionalities of Ontologies in the Semantic Web	538
13.3.2	Applications of Ontologies in the Semantic Web	541
13.3.2.1	Semantic Search	541
13.3.2.2	Semantic Portals	546
13.3.2.3	Semantic Information Integration	547
13.3.2.4	Intelligent Advisory Systems	552
13.3.2.5	Semantic Middleware	554
13.3.2.6	Semantic Software Engineering	556
13.3.3	Selected Application Domains	558
13.3.3.1	Ontologies About Cultural Heritage	558
13.3.3.2	Ontologies in eGovernment	559
13.3.3.3	Ontologies in the Life Science Domain	559
13.4	<i>Related Resources</i>	561
13.5	<i>Future Issues</i>	562
13.6	<i>Cross-References</i>	564

Abstract: Ontologies have become a prominent topic in Computer Science where they serve as explicit conceptual knowledge models that make domain knowledge available to information systems. They play a key role in the vision of the Semantic Web where they provide the semantic vocabulary used to annotate websites in a way meaningful for machine interpretation. As studied in the context of information systems, ontologies borrow from the fields of symbolic knowledge representation in Artificial Intelligence, from formal logic and automated reasoning and from conceptual modeling in Software Engineering, while also building on Web-enabling features and standards.

Although in Computer Science ontologies are a rather new field of study, certain accomplishments can already be reported from the current situation in ontology research. Web-compliant ontology languages based on a thoroughly understood theory of underlying knowledge representation formalisms have been and are being standardized for their widespread use across the Web. Methodological aspects about the engineering of ontologies are being studied, concerning both their manual construction and (semi)automated generation. Initiatives on “linked open data” for collaborative maintenance and evolution of community knowledge based on ontologies emerge, and the first semantic applications of Web-based ontology technology are successfully positioned in areas like semantic search, information integration, or Web community portals.

This chapter will present ontologies as one of the major cornerstones of Semantic Web technology. It will first explain the notion of formal ontologies in Computer Science and will discuss the range of concrete knowledge models usually subsumed under this label. Next, the chapter surveys ontology engineering methods and tools, both for manual ontology construction and for the automated learning of ontologies from text. Finally, different kinds of usage of ontologies are presented and their benefits in various application scenarios illustrated.

13.1 Scientific and Technical Overview: Foundations

Ontologies gained momentum in Computer Science in the recent years, providing a tool for the explicit representation of knowledge that is otherwise implicitly captured in the software that implements an information system. Often, it is beneficial to have such explicit knowledge about the application domain available for the information system to interact with it at runtime and to share it with other software systems. The mechanisms around ontologies allow for such an interaction with and the sharing of explicitly represented domain knowledge.

This section presents the basics of ontologies as used in Computer Science. After clarifying the notion of ontologies, it elaborates on their use for information systems and reviews various types of ontologies.

13.1.1 Notion of Ontologies

Originating from Philosophy, the notion of ontology has found its way into the field of Computer Science, where ontologies are viewed as conceptual yet computational knowledge

models. The notion of ontologies as such will be investigated, giving a definition, identifying some essential characteristics and presenting a formal ontology model.

13.1.1.1 Origin and Definition

In this chapter, the uncountable form “ontology” with its origin in philosophy and the countable variant “an ontology” as used in Computer Science, following [1], will be distinguished.

Ontology

In its original meaning in philosophy, *ontology* is a branch of metaphysics and denotes the philosophical investigation of existence. It is concerned with the fundamental question of “what kinds of things are there?” and leads to studying general categories for all things that exist dating back to the times of Aristotle [2].

Transferred to knowledge representation and Computer Science, information systems can benefit from the idea of ontological categorization. When applied to a limited domain of interest in the scope of a concrete application scenario, ontology can be restricted to cover a special subset of the world. Examples of ontological categories in, for example, the technical vehicular domain are “Vehicle,” “Car,” or “Engine.” In this sense, ontology provides a semantic vocabulary to define the meaning of things.

Ontologies

While “ontology” studies what exists in a domain of interest, “an ontology” is a computational artifact that encodes knowledge about this domain in a machine-processable form to make it available to information systems. In various application contexts, and within different communities, ontologies have been explored from different points of view, and there exist several definitions of what an ontology is. Within the Semantic Web community the dominating definition of *an ontology* is the following, based on [3].

Definition 1 (ontology). *An ontology is a formal explicit specification of a shared conceptualization of a domain of interest.*

This definition captures several characteristics of an ontology as a specification of domain knowledge, namely, the aspects of formality, explicitness, consensus, conceptuality, and domain specificity, which require some explanation.

- *Formality* – An ontology is expressed in a knowledge representation language that is based on the grounds of formal semantics. This ensures that the specification of domain knowledge in an ontology is machine-processable and is being interpreted in a well-defined way. The techniques of symbolic knowledge representation, typically built on the principles of logic, help to realize this aspect.
- *Explicitness* – An ontology states knowledge explicitly to make it accessible for machines. Notions that are not explicitly included in the ontology are not part of

the machine-interpretable conceptualization it captures, although humans might take them for granted by common sense.

- *Consensus* – An ontology reflects an agreement on a domain conceptualization among people in a community. The larger the community, the more difficult it is to come to an agreement on sharing the same conceptualization. In this sense, the construction of an ontology is associated with a social process of reaching consensus.
- *Conceptuality* – An ontology specifies knowledge in a conceptual way in terms of conceptual symbols that can be intuitively grasped by humans, as they correspond to the elements in their mental models. (In contrast to this, the weights in a neural network or the probability measures in a Bayesian network would not fit this notion of conceptuality.) Moreover, an ontology describes a conceptualization in general terms and does not only capture a particular state of affairs. Instead of making statements about a specific situation involving particular individuals, an ontology tries to cover as many situations as possible that can potentially occur [4].
- *Domain Specificity* – The specifications in an ontology are limited to knowledge about a particular domain of interest. The narrower the scope of the domain for the ontology, the more an ontology engineer can focus on capturing the details in this domain rather than covering a broad range of related topics.

In summary, an ontology used in an information system is a conceptual yet executable model of an application domain. It is made machine-interpretable by means of knowledge representation techniques and can therefore be used by applications to base decisions on reasoning about domain knowledge.

13.1.1.2 Essential Characteristics of an Ontology

Often, ontologies are visualized and thought of as *semantic networks* that display interrelated conceptual nodes, as exemplarily depicted in [Fig. 13.1](#). There, a fragment of a conceptual model about vehicles and their parts is shown in form of a graph with conceptual nodes and arcs. Intuitively, the network captures knowledge such as “cars are

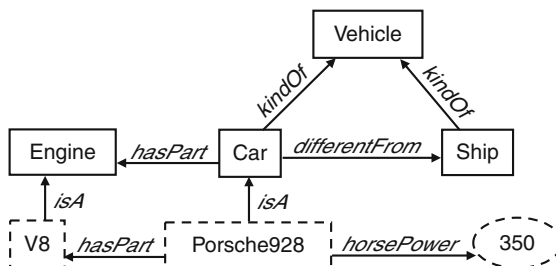


Fig. 13.1

An example ontology as semantic network

kinds of vehicles different from ships and have engines as their parts” or “a Porsche928 is a car that has a V8 engine and horsepower 350.”

Although there are various ontology languages with different semantics for similar language constructs next to a multitude of differing ontology formalizations, one can use this semantic network metaphor to identify some *essential characteristics* that are common to most ontologies used in information systems.

- *Interrelation* – Without any interconnection, the plain nodes in a semantic network like the one depicted in ▶ Fig. 13.1 would be rather a loose collection of concepts. *Relations* between concepts allow for such an interconnection, enriching the conceptual nodes with structure. In the example, cars are related to engines by means of a *hasPart* relation.
- *Instantiation* – An essential distinction is the one made between concrete individual objects of the domain of interest and more general categories that group together objects with some common characteristics. *Instantiation* is the mechanism to allow for this distinction by assigning individual objects to classes as their instances, such as the specific Porsche in ▶ Fig. 13.1 is an instance of the class of all cars. This mechanism can be found across almost all forms of conceptual models, sometimes even in the form of meta-modeling allowing for classes to be instances of (meta-) classes themselves.
- *Subsumption* – The most common way of interlinking general conceptual nodes is by *subsumption*, expressing a *kind of* relationship that reflects the notion of specialization/generalization. In the example, cars are stated to be kinds of vehicles, and thus, inherit their properties being their specialization. Subsumption is the mechanism behind the feature of inheritance hierarchies prevalent in conceptual modeling.
- *Exclusion* – A rather sophisticated means of representation is to state “negative” knowledge in form of class exclusion, preventing two general conceptual nodes from overlapping in their extensions. In the example, cars are stated to be different from ships, meaning that being a car excludes being a ship. This form of negative knowledge is found in rather expressive ontology languages that allow for negation.
- *Axiomatization* – The interrelated conceptual nodes as such are often not sufficient to express rich knowledge – they need to take part in complex statements about the domain of interest, which accounts for the notion of *axiomatization*. Besides subsumption, exclusion, or instantiation as simple forms of axiomatization, many ontology languages allow for the formulation of more complex statements in form of general *axioms*. Complex axioms are typically neglected in the graphical presentation of an ontology in lack of an appropriate paradigm for visualization.
- *Attribution* – Although not motivated by the underlying logical formalisms for knowledge representation, the inclusion of statements about datatypes and their values are a common feature in ontology languages, which accounts for the *attribution* of conceptual nodes by strings, numbers, and other data-types. In the example, cars are attributed with a number indicating their horsepower. Datatypes and values are often an indispensable feature in many applications of ontologies.

13.1.1.3 Formal Ontology Model

To present the technical constituents of an ontology in a precise way, a formal ontology model is introduced, defined in [5], that accounts for the essential characteristics identified above. This model is at the same time simple to not present auxiliary characteristics in an overly formal manner, and expressive enough to fit most of the common knowledge representation formalisms and languages used for ontologies in the Semantic Web.

Constituents of an Ontology

An *ontology* \mathcal{O} is a tuple

$$\mathcal{O} = (\mathbf{S}, \mathbf{A}) \quad (13.1)$$

of a *signature* \mathbf{S} and a set of *axioms* \mathbf{A} . At this first level of distinction, the elements in the signature, which comprise the conceptual *entities* used for knowledge representation, are separated from the actual statements that use these elements to express knowledge about the domain in the form of axioms. Hence, the entities in an ontology's signature form a semantic vocabulary to be used by knowledge engineers for the formulation of axioms. The axioms themselves are expressed in a specific ontology language or knowledge representation formalism. Often, first-order predicate logic is used as a unifying syntactic framework for expressing axioms.

The signature comprises several sets

$$\mathbf{S} = C \cup I \cup P \quad (13.2)$$

of classes C , instances I and properties P , the distinction of which is an essential feature in almost all forms of conceptual modeling ranging from UML (<http://www.uml.org/>) in software design to logically expressive ontology languages like OWL (<http://www.w3.org/TR/owl2-overview/>). Instances map to the individual nodes in a semantic network, representing individual objects of the domain of interest, such as a particular car or person. Classes map to conceptual nodes grouping together instances that have certain characteristics in common, such as the class of all things that are vehicles. Properties map to the customary arcs in semantic networks to express the relation between instances of classes, such as a car having an engine as its part. Hence, at this level of distinction, the notions of instantiation and interrelation are introduced by providing for the respective kinds of vocabulary elements to associate classes with their instances or to interrelate instances via properties. For axioms expressed in first-order predicate logic, classes correspond to unary predicates, properties to binary predicates, and instances to constant symbols, all occurring within the axioms.

The signature entities are further divided into sets

$$C = \mathcal{C} \cup \mathcal{D}, \quad I = \mathcal{I} \cup \mathcal{V}, \quad P = \mathcal{R} \cup \mathcal{T} \quad (13.3)$$

of *concepts* \mathcal{C} and *datatypes* \mathcal{D} , *individuals* \mathcal{I} , and *data values* \mathcal{V} , as well as *relations* \mathcal{R} and *attributes* \mathcal{T} . At this third level of distinction, the participants of instantiation, namely,

classes and instances, are further split, and abstract concepts are distinguished from datatypes like string or integer, while abstract individual objects of the domain are separated from concrete data values. Accordingly, properties are split into relations, which involve abstract domain objects only, and attributes of classes, which involve also data values. This reflects the natural separation of abstract domain objects from attributes of such objects, which are merely data values attached to them, and thus accounts for attribution. This distinction is also made in entity-relationship modeling in databases, in object-oriented software design, and in ontology languages like OWL, where object properties are separated from datatype properties. In this sense, individuals serve as instances of concepts, while data values serve as instances of datatypes. The explicit distinction between abstract objects of the domain and concrete data values introduced at this level can be found in most forms of conceptual modeling of Computer Science artifacts, and the presence of datatypes is important in many applications.

Specific Axioms

Based on the essential characteristics of an ontology introduced earlier, some special types of axioms are identified that are common to most ontology languages, for which a special notation in [Table 13.1](#) is introduced. Reference [5] provides mappings for these axiom types to specific ontology languages. Their intuitive meaning is as follows.

- *Instantiation* – An instantiation axiom assigns an instance to a class. The axiom α_{\wedge} (*Porsche928*, *Car*), for example, states a particular individual Porsche to be a member of the concept car.
- *Assertion* – An assertion axiom assigns two instances by means of a property. The axiom α_{\rightarrow} (*Porsche928*, *horsePower*, 350), for example, asserts the data value 350 to fill a particular Porsche's horsepower attribute.
- *Subsumption*: A subsumption axiom for two classes states that any instance of the subsumed class is also an instance of the subsuming class, while for two properties, it states that any two instances connected by the subsumed property are also connected by the subsuming one. The axiom α_{Δ} (*Car*, *Vehicle*), for example, states that any car is also a vehicle.

■ **Table 13.1**

Special axioms common in ontology formalisms

Axiom type	Notation	First-order logic expression
Instantiation	$\alpha_{\wedge}(i, C)$	$[C(i)], i \in I, C \in \mathcal{C}$
Assertion	$\alpha_{\rightarrow}(i_1, p, i_2)$	$[p(i_1, i_2)], i_1, i_2 \in I, p \in \mathcal{P}$
Subsumption	$\alpha_{\Delta}(E_1, E_2)$	$[\forall x : E_1(x) \rightarrow E_2(x)], E_1, E_2 \in \mathcal{C} \cup \mathcal{P}$
Domain	$\alpha_{D \rightarrow}(p, D)$	$[\forall x, y : p(x, y) \rightarrow D(x)], p \in \mathcal{P}, D \in \mathcal{C}$
Range	$\alpha_{\rightarrow R}(p, R)$	$[\forall x, y : p(x, y) \rightarrow R(y)], p \in \mathcal{P}, R \in \mathcal{C}$
Disjointness	$\alpha_{\oplus}(C_1, C_2)$	$[\forall x : C_1(x) \wedge C_2(x) \rightarrow \perp], C_1, C_2 \in \mathcal{C}$

- *Domain*: A domain axiom for a property and a class states that for any connection of two instances by that property the source element is an instance of the domain class. The axiom $\alpha_{D \rightarrow}$ (*horsePower*, *Car*), for example, states that the horsepower attribute has domain car.
- *Range*: A range axiom for a property and a class states that for any connection of two instances by that property the target element is an instance of the range class. The axiom $\alpha_{\rightarrow R}$ (*horsePower*, *Integer*), for example, states that the horsepower attribute has range integer.
- *Disjointness*: A disjointness axiom for two classes states that no instance of the one class can also be an instance of the other class, and thus, the classes exclude each other. The axiom α_{\oplus} (*Car*, *Ship*), for example, states the concepts for cars and ships to be disjoint.

Although in general, most ontology languages allow for the formulation of arbitrarily complex and sophisticated axioms to express domain knowledge, this set of basic axioms covers most of what is typically found in the axiomatizations of prevalent Semantic Web ontologies.

13.1.2 Ontologies in Information Systems

Ontologies in information systems build on symbolic knowledge representation techniques known from Artificial Intelligence, and they are used to access domain knowledge provided by means of Web-compliant languages. Next, ontologies will be related to knowledge representation and reasoning and an overview on ontology languages will be given.

13.1.2.1 Ontologies and Formal Knowledge Representation

An ontology – or more precisely its set of axioms – is often seen as a *knowledge base* maintained by a knowledge-based system to have access to and reason about domain knowledge.

Knowledge Representation

Knowledge representation and reasoning aim at designing computer systems that reason about a machine-interpretable representation of the world. Knowledge-based systems have a computational model of some *domain* of interest – their knowledge base *KB* – in which symbols serve as surrogates for real-world domain artifacts, such as physical objects, events, relationships, etc. [6]. The domain of interest can cover any part of the real world or any hypothetical system about which one desires to represent knowledge for computational purposes, while reasoning is performed by manipulating the symbols in the knowledge base. Hence, knowledge representation appears to be an appropriate means for realizing the notion of ontologies in computer systems according to Definition 1.

In contrast to methods of nonsymbolic Artificial Intelligence (like connectionism or statistical machine learning), symbolic knowledge representation builds on processing explicitly modeled pieces of knowledge that are represented in a well-structured way. Therefore, it usually goes along with the manual task of knowledge engineering, and the carefully designed content of a knowledge base is typically of high quality with no noise in data. Alternative approaches that skip manual engineering effort rather rely on a sophisticated runtime interpretation of natural language text or on the automated learning of classifiers from sufficiently large corpora of sample data.

A distinction often made between a knowledge base and an ontology is to see the knowledge base capturing information about a particular state of affairs in the domain, while the ontology captures more general information about any possible situation, as, for example, described in [1]. In this sense, the term ontology is often used for referring to schema knowledge about the classes C and their interrelations P expressed through axioms of type α_{Δ} , $\alpha_{D \rightarrow}$, $\alpha_{\rightarrow R}$ or α_{\oplus} , whereas a knowledge base rather comprises plain facts about the concrete instances I expressed through axioms of type α_{\wedge} and α_{\rightarrow} . However, various expressive ontology languages blur such a clear distinction by allowing for all kinds of axioms in what they call the specification of an ontology. Therefore, one can see a knowledge base also as a technical means for working with knowledge, in which a knowledge-based system loads (parts of) the specification of an ontology, most likely together with other pieces of knowledge, to take it into account for reasoning. Technically, this amounts to interpreting an ontology's axioms as a knowledge base, that is, $KB = A$, when reasoning about the domain knowledge it captures.

Reasoning

In knowledge-based systems, the notion of reasoning is associated with the process of reaching conclusions. The axioms that are contained in a knowledge base constitute the *explicit knowledge* a system has about the domain of interest, while the ability to process explicit knowledge computationally by means of reasoning allows the system to derive *implicit* knowledge that logically follows from what has been stated explicitly.

Due to the highly structured form of representation, symbolic approaches to knowledge representation allow for reasoning based on formal logic, which is a powerful tool to simulate the process of reaching conclusions. Logic provides the means to precisely determine what follows from a set of axioms based on formal semantics. Two important aspects of logic-based reasoning with ontologies in the Semantic Web are mainly looked at, namely, the *verification* of an ontology's specification and the *deduction* of new axioms.

- *Verification* – To ensure that an ontology is a good representation of its domain of discourse, reasoning can be used to validate the entirety of axioms in the respective knowledge base at least for their technical soundness. An ontology that contains contradictory information is not considered to be a good domain representation. Reasoning can be used to detect erroneous modeling in an automated way and to report it to knowledge engineers. Erroneous modeling checked for by verification typically comprises logical contradictions or a violation of explicitly stated constraints.

- *Deduction* – Based on the assumption that an ontology correctly represents the domain of interest, the process of deduction derives implicit conclusions that hold in any situation coherent with the axioms in the respective knowledge base, capturing the notion of *logical consequence*. Reasoning constitutes the primary mechanism to determine the deductive conclusions from the symbols and structure of axioms in a knowledge base. The form of statements derived as deductive consequences can thereby range from simple facts to complex generic axioms. An axiom α following from a knowledge base KB is typically denoted by $KB \models \alpha$ using the entailment symbol.

The basic operations a knowledge-based system can perform on its knowledge base are usually called `tell` and `ask` [7]. The `tell`-command adds a new statement to the knowledge base, whereas the `ask`-command is used to query what is known. While the `tell`-command determines the explicit knowledge, the `ask`-command operates under deductive closure; that is, it also gives answers that constitute implicit knowledge found by reasoning. In this sense, the command `tell(KB, α)` performs the operation $KB := KB \cup \alpha$ on a knowledge base KB and an axiom α , while the command `ask(KB, α)` yields true if and only if α is entailed by KB , i.e., $KB \models \alpha$.

13.1.2.2 Usage of Ontologies

Ontologies are used by information systems to be queried for domain knowledge. Although they seem to be very similar to other conceptual models, their ability to access implicit knowledge renders them as distinct artifacts used for building intelligent computer systems.

Ontologies Versus Other Conceptual Models

At a first glance, ontologies seem to be very similar to other kinds of conceptual models used in Computer Science, especially when looking at the typical constructs of established ontology languages. Notions such as interrelation, instantiation, or subsumption can also be found in, for example, UML class diagrams for technical specifications of information systems or in entity-relationship diagrams for the specification of database schemas. However, there is a subtle difference between ontologies and these other forms of conceptual models that is primarily motivated in terms of usage and purpose. Namely, conceptual models known from software and database engineering are prescriptive in that they are means to construct a technical system anew, whereas ontologies are descriptive capturing the knowledge observed to characterize a domain.

The main purpose and goal of designing software systems using UML class diagrams is to prescribe the technical components of an information system that is to be run on a (physical) computer system. Once the system runs, the conceptual model behind it has fulfilled its function and is not consulted or modified at runtime.

A similar argument holds true for conceptual models in database management systems: Entity-relationship diagrams as a conceptual model are often used only at *system-design time* as a preliminary step before designing the relational schema as

a basis for efficient storage and data access. They are not intended to be used or changed when the information system is in use.

In contrast to these other conceptual models, the primary purpose of an ontology is to serve as a source of domain knowledge to be queried by an information system that bases *runtime* decisions on the respective answers. By means of performing reasoning, these answers include implicit as well as explicit knowledge. This difference typically imposes additional requirements on the expressivity of knowledge representation formalisms and ontology languages, since the task of representing domain knowledge as such is very general and not restricted to specific aspects of processing, such as imperative runtime behavior or efficient data access. More expressive modeling constructs like negation or class exclusion are, therefore, typically not found outside ontology engineering.

Reasoning with Ontologies

An information system interacts with an ontology $\mathcal{O} = (\mathbf{S}, \mathbf{A})$ via the `tell-/ask`-interface. Whenever it encounters new knowledge, it applies the `tell`-command to the set of axioms \mathbf{A} , which might also introduce new symbols in the signature \mathbf{S} . When querying for knowledge, it makes use of the `ask`-command, which answers under deductive closure of \mathbf{A} seen as a knowledge base. For asking, the two reasoning tasks of verification and deduction apply.

Verification – Applied to ontologies, the reasoning task of verification checks whether the set of axioms \mathbf{A} forms a logically sound specification free of contradictory information and over-constrained restrictions on an ontology’s entities. As an example, consider the ontology given in [▶ Table 13.2](#). Here, the restrictions on the individual *SportiveMinibus* in \mathbf{S}_1 are over-constrained by the axioms in \mathbf{A}_1 , yielding a contradiction: On the one hand, the sportive minibus is stated to be a sports car; on the other hand, it is also stated to be van, while all vans are family cars; hence, the sportive minibus is also a family car, and since family cars are disjoint from sports cars, this leads to a logically inconsistent situation.

Deduction – The reasoning task of deduction allows for drawing conclusions from an ontology’s specification, thus providing access to the implicit knowledge captured in its axioms. As an example, consider the ontology given in [▶ Table 13.3](#). Here, various

■ **Table 13.2**

Example of a contradictory ontology

Ontology $\mathcal{O}_1 = (\mathbf{S}_1, \mathbf{A}_1)$	
Signature \mathbf{S}_1	
$\mathcal{C}_1 = \{\text{Van}, \text{FamilyCar}, \text{SportsCar}\}, \mathcal{I}_1 = \{\text{SportiveMiniBus}\}$	
Axioms \mathbf{A}_1	
$\alpha_{\Delta}(\text{Van}, \text{FamilyCar})$	Vans are family cars
$\alpha_{\oplus}(\text{FamilyCar}, \text{SportsCar})$	Family cars and sports cars are different things
$\alpha_{\wedge}(\text{SportiveMinibus}, \text{Van})$	The sportive minibus is a van
$\alpha_{\wedge}(\text{SportiveMinibus}, \text{SportsCar})$	The sportive minibus is a sports car

Table 13.3

An example ontology for deduction

Ontology $\mathcal{O}_2 = (\mathbf{S}_2, \mathbf{A}_2)$	
Signature \mathbf{S}_2	
$\mathcal{C}_2 = \{\text{SportsCar}, \text{Car}, \text{Vehicle}, \text{Engine}\}, \mathcal{I}_2 = \{\text{Porsche928}, \text{V8}\}, \mathcal{R}_2 = \{\text{hasEngine}\}$	
Axioms \mathbf{A}_2	
$\alpha_{\Delta}(\text{SportsCar}, \text{Car})$	Sports cars are cars
$\alpha_{\Delta}(\text{Car}, \text{Vehicle})$	Cars are vehicles
$\alpha_{\rightarrow R}(\text{hasEngine}, \text{Engine})$	hasEngine ranges over engines
$\alpha_{\wedge}(\text{Porsche928}, \text{SportsCar})$	The Porsche928 is a sports car
$\alpha_{\rightarrow}(\text{Porsche928}, \text{hasEngine}, \text{V8})$	The Porsche928 has a V8 engine

conclusions can be drawn from the explicitly stated axioms when, for example, a first-order logic semantics is assumed. Since the individual *Porsche928* is stated to be a sports car and sports cars are stated to be special kinds of cars, it can be concluded to also be a car, that is, $\mathbf{A}_2 \models \alpha_{\wedge}(\text{Porsche928}, \text{Car})$. And since cars are stated to be special kinds of vehicles, it can furthermore be concluded to be a vehicle as well, that is, $\mathbf{A}_2 \models \alpha_{\wedge}(\text{Porsche928}, \text{Vehicle})$. As for conclusions at class level, it can be inferred that any sports car is also a vehicle, that is, $\mathbf{A}_2 \models \alpha_{\Delta}(\text{SportsCar}, \text{Vehicle})$. Moreover, since the range of the property *hasEngine* is stated to be the class *Engine*, the V8 engine, which is assigned to be the engine of a particular car, can be concluded to be an engine, that is, $\mathbf{A}_2 \models \alpha_{\wedge}(\text{V8}, \text{Engine})$. In summary, querying an ontology under the mechanism of deduction yields more knowledge than that explicitly stated due to reasoning.

13.1.2.3 Ontology Languages

In the recent years, a landscape of specialized languages for expressing ontologies has evolved, mostly in the context of Semantic Web research. Besides covering the essential characteristics of an ontology as a knowledge representation artifact, they also account for Web-enabling features such as a unique identification of entities via URIs or XML serialization formats. The most prevalent of these Semantic Web ontology languages will be surveyed briefly.

RDF(S) – As an effort for the standardization of metadata, the Resource Description Framework **RDF** [8] and a simple ontology language **RDFS** (RDF Schema) [9] emerged early in the context of the Semantic Web as an initiative from the World Wide Web Consortium (W3C, <http://www.w3.org/>). Meanwhile, **RDF(S)** has become a well-established and widely accepted standard for encoding metadata and basic ontologies on the Web (see also [Semantic Annotation and Retrieval: RDF](#)).

As an ontology language, RDFS has the expressivity for the features of interrelation, instantiation, and subsumption, by hierarchies that can be built over resource classes and

properties. It also allows for attribution in reference to XSD datatypes but restricts axiomatization to domain and range restrictions besides subclassing and typing. In particular, RDFS does not exhibit the feature of expressing exclusion or negation of any form, which renders it as a semantically rather lightweight formalism.

OWL – On top of RDF(S), W3C standardization efforts have produced the Web Ontology Language (**OWL**) [10] as the currently most prominent language to express ontologies for their use in the Web. OWL comes in several variants with increasing expressiveness, of which only the most expressive one, namely **OWL Full**, has a proper layering on top of RDF(S), allowing for features of meta-modeling and reification, while the variant **OWL DL** is focused on the formalism of description logic (DL) [11]. In its newest version **OWL 2**, the Web Ontology Language has recently undergone the final steps of W3C standardization.

Besides the class membership and subsumption relations inherited from RDF(S), OWL offers the construction of complex classes from simpler ones by means of logical expressions, which accounts for rich axiomatization including class exclusion. The design goals behind OWL are primarily driven by expressive description logics as decidable fragments of first-order predicate logic (see also [► KR and Reasoning on the Semantic Web: OWL](#)).

Rule Languages – Besides ontologies as characterized by the W3C standard languages, there is the logic programming paradigm of knowledge representation based on *rules* with an if-then reading, which accounts for a special form of axiomatization. Rule-based systems and deductive databases have brought forth specific languages for expressing rules, such as **F-Logic** (Frame Logic) [12] or **Datalog** [13]; however, these are not tailored to Web standards. An attempt to integrate rules with Web ontologies is the Semantic Web Rule Language (**SWRL**) as a W3C member submission to build rules into OWL ontologies. Suggestions for a restricted use of SWRL-style rules on top of OWL are, for example, DL-Safe Rules [14] or DL Rules [15]. Rule languages for the Semantic Web are further discussed in W3C's Rule Interchange Format (RIF) Working Group with the goal to establish **RIF** as a Web standard for rule languages complementing RDF(S) and OWL (see also [► KR and Reasoning on the Semantic Web: RIF](#)).

WSML – The Web Service Modeling Language (**WSML**) [16] is an attempt to standardize ontology languages for the Web with a special focus on annotating Semantic Web Services [17]. Next to the service-specific aspects, it provides expressive means for the formulation of ontologies in general, addressing various knowledge representation paradigms in different language variants (see also [► Semantic Web Services](#)).

Semantic Vocabularies – Apart from expressive knowledge representation formalisms, there are also languages to form controlled semantic vocabularies, which can be used to formulate specific lightweight ontologies with rather few but easy-to-use semantic features. One such effort is the Simple Knowledge Organization System (**SKOS**) [18], which is built on top of the RDF-based W3C standards. Another lightweight approach for expressing semantic networks on the Web is the ISO-standard **Topic Maps** with a focus on a graph-oriented paradigm of representing and visualizing knowledge.

13.1.3 Variants of Ontologies

Since ontologies in Computer Science are a rather new phenomenon, the notion of an ontology is perceived in a broad sense. Many different forms of conceptual models are arguably interpreted as ontologies, varying in different dimensions. Varying forms of ontology appearance, scope, and degree of formality will be discussed next.

13.1.3.1 Varying Form of Appearance

When engineered for or processed by information systems, ontologies appear in different forms. A knowledge engineer views an ontology by means of some graphical or formal visualization, while for storage or transfer, it is encoded in an ontology language with some machine-processable serialization format. A reasoner, in turn, interprets an ontology as a set of axioms that constitute a logical theory.

Graphical Appearance

In knowledge engineering tools, an ontology is often visualized as some form of semantic network with interlinked conceptual nodes. [▶ Figure 13.1](#) gives an impression of such a graphical visualization. Most typical for such a visualization is to display the taxonomic hierarchy of domain concepts and the customary relations between them. However, certain information, such as the knowledge captured in complex axioms, cannot easily be displayed graphically because of the lack of appropriate visualization paradigms.

Formal Appearance

As ontology languages like OWL are based on logical formalisms, the formal semantics of the language precisely defines the meaning of an ontology in terms of logic. To a reasoner, therefore, an ontology appears as a set of logical formulas that express the axioms of a logical theory. The following logical formulas constitute a set of axioms that formalize the knowledge from the semantic network in [▶ Fig. 13.1](#) in the description logic formalism that underlies, for example, the language OWL.

$$\begin{aligned}
 & \dots \\
 & \textit{Car} \sqsubseteq \textit{Vehicle} \\
 & \textit{Ship} \sqsubseteq \textit{Vehicle} \\
 & \textit{Car} \sqcap \textit{Ship} \sqsubseteq \perp \\
 & \textit{Car} \sqsubseteq \forall \textit{hasPart.Engine} \\
 & \exists \textit{hasEngine.T} \sqsubseteq \textit{Car} \\
 & \textit{Car} (\textit{Porsche928}), \textit{Engine} (\textit{V8}) \\
 & \textit{hasEngine} (\textit{Porsche928}, \textit{V8}), \textit{horsePower} (\textit{Porsche928}, 350) \\
 & \dots
 \end{aligned}$$

This form of appearance of an ontology is free of syntactical or graphical additions or ambiguities and reflects the pure knowledge representation aspect.

Machine-Processable Appearance

When exported for storage on disk or for transfer over the wire, an ontology's specification needs to be expressed in a machine-processable representation format. Hence, to the developer of an ontology editor, storage facility or reasoning tool, an ontology appears in form of some serialization format suitable for machine processing. The following listing displays the information from the formal view above in the RDF XML serialization format.

```

...
<owl :Class rdf :about = " # Vehicle " />
<owl :Class rdf :about = " # Car ">
  <rdfs :subClassOf rdf :resource = " # Vehicle " />
  <rdfs :subClassOf>
    <owl :Restriction>
      <owl :onProperty rdf :resource = " # hasEngine " />
      <owl :allValuesFrom rdf :resource = " # Engine " />
    </ owl :Restriction>
  </ rdfs :subClassOf>
  <owl :disjointWith rdf :resource = " # Ship " />
</ owl :Class>
<owl :Class rdf :about = " # Ship ">
  <rdfs :subClassOf rdf :resource = " # Vehicle " />
</ owl :Class>
<owl :Class rdf :about = " # Engine " />
<owl :ObjectProperty rdf :about = " # hasEngine ">
  < rdfs :domain rdf :resource = " # Car " />
</ owl :ObjectProperty>
<owl :DatatypeProperty rdf :about = " # horsePower " />
<Car rdf :about = " # Porsche 9 2 8 ">
  <horsePower> 3 5 0 < / horsePower>
  <hasEngine rdf :resource = " # V 8 " />
</ Car>
<rdf :Description rdf :about = " # V 8 " />
...

```

It shows the various interconnected concepts, properties, and individuals in their technical XML rendering to be parsed by ontology tools that can read OWL in its RDF XML serialization.

13.1.3.2 Varying Scope

As the domain of an ontology can cover any topic that suits conceptual modeling, there is no restriction to what kind of knowledge can be represented. However, ontologies have been classified in different types according to the kind of knowledge they capture.

The most established distinction is between so-called *upper-level ontologies*, which describe very general categories that can be used across domains, and *domain ontologies*, which capture the knowledge of a specific domain.

Upper-Level Ontologies

Upper-level ontologies attempt to describe very abstract and general concepts that can be shared across many domains and applications. Upper-level ontologies are sometimes also called top-level ontologies or foundational ontologies. They borrow from philosophical notions, describing top-level concepts for all things that exist, such as “physical object” or “information object,” as well as generic notions of common-sense knowledge about phenomena like time, space, processes, etc. They are usually well thought out and extensively axiomatized.

Due to their generality, upper-level ontologies are typically not directly used for conceptual modeling in applications but reused as a basis for building more specific ontologies. For a concrete application, it would in most cases not make sense to directly use an upper-level ontology as it is free of any domain knowledge. Often, upper-level ontologies are employed to merely guide the design of domain ontologies, prescribing various knowledge representation patterns that can be instantiated.

Domain Ontologies

Domain ontologies capture the knowledge within a specific domain of discourse, such as medicine or geography. In this sense, they have a much narrower and more specific scope than upper-level ontologies. Prominent ontologies exist, for example, in natural sciences, such as medicine, genetics, geographic and communal efforts such as environment information, tourism, as well as cultural heritage and museum exhibits, to name just a few.

Domain ontologies can be used together with upper-level ontologies in a combined way. In such a case, the domain ontology is typically aligned with the upper-level ontology, meaning that its specific domain concepts subclass or instantiate broader top-level notions inheriting their upper-level axiomatization.

13.1.3.3 Varying Degree of Formality

Ontologies used in the context of the Semantic Web can also be distinguished according to their degree of formality. An ontology’s degree of formality determines to which extent it is axiomatized by means of logical statements about the domain. *Lightweight* ontologies possess no or only a few axioms constraining the use of the entities in their signature. On the other hand, *heavyweight* ontologies are characterized by extensive axiomatization for interrelating the signature elements in a sophisticated way – such that nearly all entities are accompanied by many axioms constraining their use and supporting reasoning about them.

Thesauri

The least formal form of an ontology are Thesauri, which organize the words used in a certain domain according to lexical criteria. Examples are language-specific dictionaries

that also encode information about synonyms to be used in word-processing software, or a classification of medical terms for diseases. The expressivity of Thesauri is rather low in terms of logic-based knowledge representation and is typically restricted to lexical relations between words, such as synonymy or homonymy. One of the most well-known and comprehensive general-purpose thesauri is **WordNet** [19].

Concept Schemes

As informal semantic networks of interlinked conceptual nodes, concept schemes often evolve as the result of collaborative tagging activities in a Web context. Examples are tag taxonomies [20] and informal hierarchies, for example, modeled in SKOS. They are of low expressivity, offering rather informal semantic relations and means for classification with very limited possibilities for axiomatization. This makes them well-suited to rather uncontrolled environments of collaborate editing and maintenance within a larger community of uncoordinated knowledge contributors.

Taxonomies

Taxonomies are often used for a formalized hierarchical organization of domain knowledge by means of class hierarchies based on the notion of subsumption. An example is a catalog of product categories that builds up a strict subsumption hierarchy of product classes. The main feature of taxonomies is their strict hierarchical categorization of classes. Therefore, the subsumption relationship is typically formalized logically, for example, in terms of transitivity, while no or only few other cross-relations between classes are allowed.

Conceptual Data Models

In Computer Science, the use of various conceptual models tailored toward designing information systems or database management systems is ubiquitous. Examples are entity-relationship diagrams or UML diagrams used for domain modeling. Such conceptual models typically have the expressivity for structuring a domain for the data used within a software system by means of concept subsumption hierarchies and properties and attributes of domain classes. If any logical formalization occurs, it is typically used for checking constraints on the conceptual model to identify faulty data situations rather than for drawing conclusions out of explicit knowledge.

Rule and Fact Bases

In many applications, rule or fact bases serve as data-intensive knowledge bases built for the handling of large numbers of individuals with some basic reasoning. Examples are logic programming rule bases for the derivation of instantiation and assertion axioms, or description logic A-Boxes and RDF(S) graphs for querying facts with simple reasoning over class and property hierarchies. These ontologies typically have the expressivity for interrelating and typing instances and for a rule-based derivation of facts by means of logic programming mechanisms.

General Logical Theories

The most formal and expressive type of an ontology is a general logical theory, in which the domain of discourse is highly axiomatized in an expressive logic-based knowledge representation formalism, such as first-order predicate logic or even higher-order or modal logics. An example is the formal specification of an upper-level ontology with a rich axiomatization for very general notions in the form of modal logic axioms. A general logical theory is not restricted to the derivation of facts at the instance level, but also captures a rich axiomatization about classes and properties at the schema level, allowing for drawing conclusions about general situations in the domain in the form of complex axioms.

13.1.3.4 Some Example Semantic Web Ontologies

In different areas of the Semantic Web particular ontologies have evolved fitting in different categories of the ones reported above.

Upper-Level Ontologies

As for upper-level ontologies, the **Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE)** [21] is the most prominent approach based on philosophical notions and with a representation in OWL. Another established top-level ontology is the **Suggested Upper Merged Ontology (SUMO)** [22] with a special link to the WordNet lexicon.

Catalog Schemes and eBusiness Ontologies

For the unifying categorization of materials, products, and services across enterprises, particular coding scheme standards have been established. Two such efforts are the **United Nations Standard Products and Services Code (UNSPC)** for use across enterprises throughout the global eCommerce marketplace focused on the American economy, and the similar but more expressive **eCl@ss** standard. There have been efforts to express these product categorization schemes in ontology languages like OWL, as reported in [23]. Based on this, the **GoodRelations** lightweight ontology can be used for annotating eBusiness offerings on the Web, as a non-toy vocabulary for describing the types of goods as well as terms and conditions for buying items and services offered on the Web [24].

Medical Domain Ontologies

The medical domain is one for which many ontologies are available for use in a Web context. **GALEN** is an ontology for clinical information whose encoding in OWL is widely used. The **Systematized Nomenclature of Medicine (SNOMED)** is a computer processable collection of medical terminology covering most areas of clinical information such as diseases, findings, procedures, microorganisms, pharmaceuticals, etc., expressed in the description logic formalism. Also the **Foundational Model of Anatomy (FMA)** is a medical ontology concerned with the representation of classes or types and relationships necessary for the symbolic representation of the phenotypic structure of the human body. The **International Classification of Diseases (ICD)** is a taxonomy of medical terms of

diseases for use in diagnostic classification, coming with an OWL representation. The **Unified Medical Language System (UMLS)** is a compendium of many controlled vocabularies in the biomedical area. The *UMLS Meta-thesaurus* comprises over one million biomedical concepts and five million concept names, all of which stem from one of the more than 100 incorporated controlled vocabularies and classification systems (including SNOMED, ICD, MeSH – the Medical Subject Headings), the Gene Ontology, and many more. It provides a mapping structure among the source vocabularies and thus allows one to translate among the various terminology systems; the *UMLS Semantic Network* adds a basic semantic structure by associating to each concept one out of 135 so-called *semantic types* (like, e.g., organisms, anatomical structures, biologic function, chemicals, etc.) and also incorporates additional semantic relationships. Finally, the *UMLS SPECIALIST Lexicon* further provides facilities for natural language processing by adding further lexical information.

Service Description Ontologies

For the semantic annotation of Web Services, various service-specific ontologies have been proposed. One is **OWL-S** as an effort to define an ontology for Semantic Web Services markup expressed in OWL. Another one is the **Web Service Modeling Ontology (WSMO)** [25], which is expressed in its own specific language WSML. **SAWSDL** is a W3C recommendation which defines a set of extension attributes for the Web Services Description Language (WSDL) and XML Schema definition language that allows one to describe additional semantics of WSDL components. The specification defines how semantic annotation is accomplished using references to external semantic models, for example, ontologies.

Social Web Ontologies

There is a couple of relatively simple, but very widespread RDF schemas that aim at semantically enriching information in the Social Web, making social Web information better interoperable and interpretable, thus better connecting people in the Social Web. Most importantly, **FOAF** (Friend-of-a-friend ontology) is an RDF Schema describing people, their activities, and their relations to other people and objects. Anyone can use FOAF to describe themselves. FOAF allows groups of people to build up social networks without the need for a centralized database. Related schemas are **DOAC** (Description of a Career) for including information about education, working experience, publications, spoken languages, etc., **DOAP** (Description of a Project) for describing open-source software projects, and **SIOC** (Semantically-Interlinked Online Communities) for interconnecting online discussions through different channels like blogs, forums, and mailing lists. The SIOC project has developed linked data wrappers for several popular blogging engines, content management systems and discussion forums such as WordPress, Drupal, and phpBB Endnote. Based on such published information, one can, for instance, define metrics to determine social neighborhood and social reputation, which might be used in eScience [26].

Reasoner Benchmarking Ontologies

For the benchmarking of reasoner systems, specific ontologies are frequently used, while some have even specifically been constructed for this particular purpose. The **Wine ontology** (<http://www.w3.org/TR/owl-guide/wine.rdf>) is a rather expressive OWL ontology that makes use of sophisticated description logic constructs and is thus demanding for reasoners. The **Lehigh University Benchmark** (LUBM, <http://swat.cse.lehigh.edu/projects/lubm/>) [27] and the **University Ontology Benchmark** (UOBM) [28] define specific test ontologies in the university domain that are widely used. Also some of the ontologies mentioned above are often used for testing reasoner performance due to their size or complexity, such as GALEN or SNOMED.

13.2 Scientific and Technical Overview: Engineering and Methodological Aspects

Ontologies used in various applications differ, for instance, in terms of scope, size, or expressivity, that is, their degree of axiomatization. While it is often possible to reuse existing ontologies that fulfill all the requirements of a certain application, many practical scenarios demand for the acquisition of new ontological knowledge or the adaptation of previously given axiomatizations. Additional changes to the ontology might become necessary at runtime as the domain knowledge or user requirements evolve.

The corresponding modeling and maintenance tasks make high demands on scarce expert resources and the capabilities of human ontology engineers. In order to overcome this knowledge acquisition bottleneck, the manual ontology construction process must be supported by efficient software tools, including *ontology editors* or ontology development environments [29]. Ideally, these tools should be complemented by appropriate *ontology engineering methodologies* [30] – guidelines and best practices in ontology design, developed from practical experiences as well as from theoretical considerations of formal ontology. Such methodologies are indispensable in order to prevent modeling errors that might hinder the applicability of the engineered ontologies. Furthermore, by structuring the otherwise undirected ontology engineering process and hence preventing unnecessary iterations or overly long meaning negotiations, they can significantly speed up ontology construction. Nevertheless, the construction and refinement of ontologies remains a tedious and time-consuming endeavor unless it is assisted by automatic or semiautomatic knowledge acquisition methods. So-called *ontology learning* techniques aim at acquiring ontological entities and axioms from various kinds of data, including natural language text [31], multimedia documents [32, 33], and folksonomies [34].

This section will give an overview of tools and methodologies for ontology engineering, thereby focusing on description logics and, in particular, the Web Ontology Language OWL.

13.2.1 Manual Ontology Construction

The manual construction of ontologies is a challenging and time-consuming endeavor. According to [35], ontology engineering is “*the set of activities that concern the ontology development process, the ontology life cycle, and the methodologies, tools and languages for building ontologies.*” *Ontology editors* or *ontology development environments*, which typically offer a variety of additional features and pluggable components, can significantly speed up the ontology development process, since they relieve ontology developers from the need to care about the syntactic representation of ontologies. In this section, a brief overview will be given (1) on how ontologies are constructed in real-life application scenarios, and (2) the kind of tool support that is available to actually create all the entities and axioms an average OWL ontology consists of.

13.2.1.1 Ontology Editors

Assuming it is already known that one would like to build an OWL ontology and have a rough idea of what it should look like in terms of scope, size, and expressivity, what would be the ontology editor of one’s choice? While it is difficult to make a general recommendation, there are certain aspects, which make one or the other tool preferable in a given application setting. Some of these aspects are discussed in the following.

Visualization and Editing Paradigm

The effectiveness of ontology editors or engineering environments largely depends on the usability of the user interface and the editing or visualization paradigm it adheres to. An inappropriate type of visualization, that is, a lack of expressivity at the interface level, hinders ontology construction and might even provoke certain modeling errors such as erroneous disjointness assumptions or misinterpretations of domain-range restrictions [36]. While the visualization paradigm is typically determined by the requirements of a family of ontology languages, there is also a variety of editors for specific domains such as bioinformatics [37] or particular types of ontologies, for example, in terms of logical complexity. While tree-based visualizations, for instance, as provided by the majority of today’s ontology editors, are most suitable for editing taxonomies (see [Sect. 13.1](#)) or other lightweight types of ontologies, the specification of arbitrary axioms or rules often demands for more powerful editing functionalities. To this end, more expressive graphical notations have been proposed, such as UML diagrams [38] or text-based interfaces, for example, based on controlled natural language [39–42].

Extensibility

Many ontology development environments such as Protégé or the NeOn Toolkit (see further below), therefore, consist of a plug-in infrastructure with an underlying API that can be used for extension by custom or off-the-shelf components for various ontology engineering activities, including modularization, alignment, querying, and semantic annotation.

Reasoning Support

If an ontology is constructed for the purpose of logical inference, for example, in a data integration scenario, minor modeling errors can have a huge impact on the overall usefulness of the ontology as they might lead to wrong conclusions or even cause logical inconsistency. In order to detect and possibly remove such errors, an ontology engineer should be provided with automated support for reasoning, that is, querying and classification, as well as for inconsistency diagnosis. Furthermore, depending on the intended use of an ontology and the required level of expressivity, additional language features such as rule support can turn out to be helpful in building the ontology.

Collaborative Ontology Development

The complex and time-consuming task of constructing an ontology usually involves several people with varying roles, among them *ontology engineers* and *domain experts*. In a centralized setting, these people, affiliated to different organizations and working in different places, jointly develop one common ontology. If the ontology consists of multiple parts or modules stored on different servers, for example, in a grid, and connected by mapping axioms, this ontology is typically called a *distributed ontology*. *Collaborative ontology engineering* is supported, for example, by the DILIGENT methodology (cf. ▶ Sect. 13.2.2), semantic Wikis [43], and plug-ins for various ontology editors such as Protégé [44].

▶ Table 13.4 gives an overview of the most well-known ontology development environments for the Web Ontology Language OWL. All of these tools relieve ontology engineers from the burden of having to deal with a specific serialization format, helping them to avoid syntactic modeling flaws. However, they usually cannot prevent semantics-related errors caused, for example, by common misconceptions of the OWL semantics. In order to ensure a proper representation of the domain expert's conceptualization, guidelines and methodologies are required that assist ontology engineers in formalizing previously implicit knowledge in a meaningful way. The following ▶ Sect. 13.2.2 will therefore introduce some of the existing ontology engineering methodologies.

13.2.2 Methodologies for Ontology Engineering

An *ontology engineering methodology* is a set of procedures, guidelines, and best practices derived from real-world ontology development experiences (e.g., [45]) or theoretical

■ Table 13.4

Ontology development environments for OWL

Name	Organization	URL
Protégé	Stanford University	http://protege.stanford.edu
NeOn Toolkit	NeOn Foundation	http://www.neon-toolkit.org
Swoop	University of Maryland	http://code.google.com/p/swoop/
TopBraid Composer	TopQuadrant	http://www.topquadrant.com
SemanticWorks	Altova	http://www.altova.com

considerations of formal ontology [46]. It supports ontology engineers and domain experts, for example, in defining the structure of the ontology or in selecting formal and informal data sources to be reused. In recent years, the interest in such ontology engineering methodologies has grown significantly, not only thanks to the more and more widespread use of ontologies in practical applications, but most notably also because of the increasing size and complexity of those ontologies. Constructing large ontologies of sufficient quality with a minimum of human and financial resources poses one of the greatest challenges to the vision of the Semantic Web.

In the following, we will outline the basic structure of a methodological ontology construction process and describe some of the most well-known guidelines for proper ontology design.

13.2.2.1 Generic Methodology

The core of each ontology engineering methodology essentially comprises three steps [35, 47], which are explained in the following:

- *Requirements Analysis* – Usually, an ontology engineering process starts with a detailed analysis of the requirements that arise from the underlying application scenario. The ontology engineer or domain expert describes these requirements by means of an *ontology requirements specification* document, which serves as a basis for subsequent modeling activities and quality assurance, that is, later checks against the specification. For this purpose, the description of the requirements should contain information, for example, about the scope of the ontology (*competency questions*), its intended use, or the level of expressivity.
- *Conceptualization* – In the conceptualization phase, the ontology's content is represented in terms of semantic vocabulary and statements about the target domain of interest, which involves the choice of ontological entities and the formulation of axioms. Based on the aforementioned requirements specification, ontology engineers and domain experts try to achieve a common agreement upon the basic structure of the ontology, for example, by exchanging arguments to support their respective design decisions [48]. The result of this phase is an informal or semiformal specification of their shared conceptualization.
- *Implementation* – The explicit formalization of this specification in terms of a concrete ontology representation language is the final step of the core ontology engineering methodology. Choosing an appropriate ontology language most notably depends on the intended use of the ontology and the required level of expressivity. As will be seen in [Sect. 13.2.3](#), the implementation phase can be supported by automatic approaches to ontology acquisition and reuse.

Furthermore, ontology development activities such as ontology *evaluation* [49], *versioning*, *documentation* or the reuse of existing, formal or informal resources (*knowledge acquisition*) might be required. In particular, it is often advisable to perform an alignment of the ontology with an existing upper-level ontology, whose axiomatization of the most fundamental ontological distinctions is often perceived as a valuable help in the overall quality assurance process.

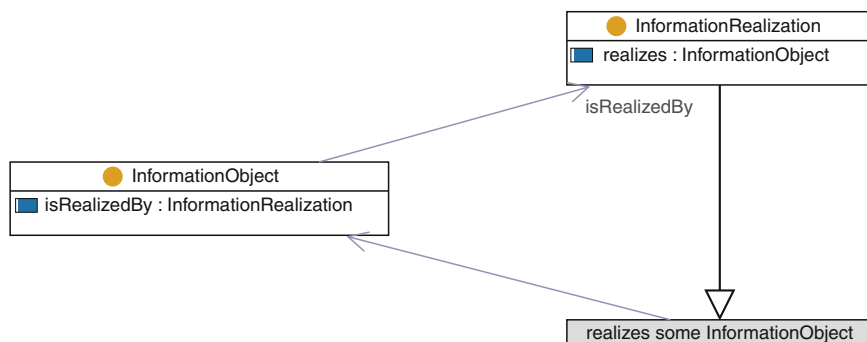
Such an alignment can be achieved, for example, by applying *ontology design patterns* [50]. Ontology design patterns, similar to their counterparts in software engineering, incorporate best practices in ontology development by providing ontology engineers with templates for specific modeling tasks. ▶ *Figure 13.2* shows a simple content pattern [51] extracted from the DOLCE upper-level ontology [21], which can be used to represent the realization of an information object (see also <http://ontologydesignpatterns.org>). It could be specialized, for example, by a class *Novel* modeling an information object and a class *Book*, which represents its physical realization. Other types of ontology design patterns include, for example, logical patterns and correspondence patterns for ontology alignment [52].

13.2.2.2 Methodologies and Tool Support

The above-mentioned steps are part of more or less every ontology engineering methodology. Some of the most well-known methodologies facilitating the effective and efficient construction of ontologies are Methontology [53], OntoClean [46], and DILIGENT [54] (see ▶ *Table 13.5*). They all have been inspired by the first, general methodologies developed in the 1990s [61, 62]. Nowadays, the state-of-the-art comprises a variety of different methodologies for specific ontology development scenarios (e.g., distributed, collaborative ontology engineering [54]), and specific application domains, such as bioinformatics [63] or medicine [45].

Only a few of these methodologies have been integrated with major ontology development environments. However, the application of OntoClean, for example, is facilitated by several plug-ins, in particular for Protégé [64], WebODE [65], and OntoEdit [66]. These plug-ins facilitate the manual tagging of ontologies with OntoClean meta-properties and provide means for checking the consistency of an ontology according to the OntoClean constraints. An automatic approach to applying OntoClean has been proposed in [67].

Despite the existence of various ontology engineering methodologies, the construction of an ontology remains a labor-intensive, time-consuming, and error-prone endeavor if it



■ Fig. 13.2

Ontology design pattern: information realization, by Valentina Presutti and Aldo Gangemi (<http://ontologydesignpatterns.org/>)

■ **Table 13.5**

Ontology engineering methodologies

Name	Institution	Reference
Methontology	Universidad Politécnica de Madrid	Fernández-López et al. [53]
Ontology Development 101	KSL, Stanford University	Noy and McGuinness [55]
On-To-Knowledge	AIFB, Karlsruhe Institute of Technology	Sure et al. [56]
OntoClean	CNR and IBM Watson	Guarino and Welty [46]
UPON	CNR and La Sapienza, Rome	De Nicola et al. [57]
DILIGENT	AIFB, Karlsruhe Institute of Technology	Tempich et al. [54]
HCOME	University of the Aegean, Samos	Kotis and Vouros [58]
NeOn Methodology	Universidad Politécnica de Madrid	NeOn Deliverable 5.4.1 [59]
DOGMA	STARLab, Vrije Universiteit Brussel	Jarrar and Meersman [60]

is carried out entirely manually. The next section will therefore take a look at some automatic methods for knowledge acquisition and ontology generation that support this manual process, which are commonly known as *ontology learning* techniques.

13.2.3 Ontology Learning

The manual construction of ontologies is a time-consuming endeavor, which makes high demands on the skills of an ontology engineer. Although various editors relieve one from the burden of dealing with the OWL syntax by providing efficient support for creating, querying, and merging ontologies, and despite all the methodological guidelines available, the *knowledge acquisition bottleneck* still hinders the widespread use of semantic technologies. A possible solution to this problem is the use of *ontology learning* methods [68].

The term “ontology learning,” coined in [68], refers to the automatic or semiautomatic generation of ontologies from various kinds of data sources including folksonomies [34], FOAF profiles [69], and multimedia documents [32, 33]. In its most commonly used sense, however, it is used to denote the acquisition of ontological knowledge from unstructured natural language text. This type of ontology learning, that could be described as “lexical” or linguistically motivated ontology learning, differs from logical approaches such as *concept learning* (ILP) or *relational exploration* (FCA) in terms of input data. While logical approaches typically derive new axioms from existing, formal, and explicit representations, lexical ontology learning has to face the challenge of dealing with large amounts of informal and often unreliable data. The following will mainly focus on the latter type of ontology learning – thereby omitting the distinction between ontology

learning and *ontology population*, which is sometimes made in order to tell apart the acquisition of schema-level knowledge from the generation of instantiations and assertions (see [Sect. 13.1](#)).

13.2.3.1 Methods for Ontology Learning from Text

The target for ontology learning can vary among certain ontological entities or types of axioms to be acquired.

Concepts

Terms, that is, nominal or verbal phrases referring to linguistic concepts, are widely accepted as a means of labeling classes, individuals, and properties. One of the most fundamental tasks in lexical ontology learning, therefore, aims to identify terms or phrases that are relevant in a particular domain of interest (*term extraction*). Often the significance of term occurrences is determined by comparing their frequencies with statistics obtained for a reference corpus [70–72], or by considering structural information that is contained in HTML or XML documents [73]. For a good overview of term extraction methods, see [74] and [75]. In a second step, based on Harris' distributional hypothesis [76], these terms can then be grouped into clusters of synonymous or otherwise related terms describing a single concept [77].

Subsumption

The backbone of any ontology is constituted by a set of taxonomic relationships among concepts (or concept descriptions). Each of the classes can be defined intentionally, for example, by a descriptive label or its relationships to other classes, as well as extensionally by specifying a set of instances belonging to this concept. Since the core taxonomy of an ontology, independently of the underlying ontology representation language, is of crucial importance for the use of ontologies as a means of abstraction, most ontology learning approaches so far have focused on the formation of concepts and concept hierarchies. Accordingly, different approaches to learning subsumption have been developed.

The vast majority of these approaches rely on lexico-syntactic patterns, invented by Marti Hearst and commonly known as “Hearst patterns” [78]. These patterns, essentially linguistic expressions encoding both lexical and syntactic constraints on the textual context of a concept expression, have shown to be a helpful means to detect *hyponymy* relationships or categories of named entities. The following pattern, for instance, would match a noun phrase (or enumeration of noun phrases) followed by “and other” and finally, a second noun phrase denoting the super-concept of the first concept expression.

"cars, ships and other vehicles"
NP { , NP } * { , } { and | or } other NP

Additional patterns have been proposed by Ogata and Collier [79], Cimiano [80], and others. For an extensive comparative evaluation of various hyponymy patterns in

a specific domain, see [81]. Common to all pattern-based approaches to hyponym extraction is the problem of data sparseness. Since occurrences of lexico-syntactic patterns are comparatively rare in natural language texts, most research nowadays concentrates on the Web as a corpus [82–85] – even though the enormous syntactic and semantic heterogeneity of Web documents poses new sorts of challenges. At the same time, approaches to the automatic generation of patterns [86–89] aim to increase the flexibility and effectiveness of ontology learning or information extraction systems. A different approach, specifically designed to support the acquisition of subsumption relationships, has been proposed by Sharon Carballo and others [90–92]. It relies on hierarchical clustering techniques in order to group terms with similar linguistic behavior into hierarchically arranged clusters.

Instantiation

The tasks of learning instantiations and assertions, often jointly referred to as *ontology population*, are targeted at the acquisition of facts, that is, instance-level information. State-of-the-art approaches to determine, for example, class membership of individuals usually build upon computational methods for named entity recognition and classification. While some of them use lexico-syntactic patterns [80] similar to the aforementioned Hearst patterns, others exploit, for example, the distributional similarity of named entities and their semantic classes [93]. A method for extracting instantiations as well as assertions from natural language text has been proposed in [94] and relies on FrameNet as a specific kind of structured background knowledge. More details on ontology population can be found in [▶ Semantic Annotations and Retrieval: Manual, Semiautomatic, and Automatic Generation](#).

Assertion

Identifying relationships between named entities or individuals in a given ontology is usually considered a subtask of information extraction. A possible distinction can be made between *open* and *closed information extraction* approaches, differing in the degree to which they rely upon supervised learning techniques. While closed information extraction (e.g., [82, 89, 95]) typically presumes an explicit or implicit specification of the target relationship, hence requiring some amount of training data or handcrafted extraction rules, open information extraction [96] relies on merely unsupervised or semi-supervised approaches.

Domain and Range

Most approaches to the acquisition of non-taxonomic relationships (or object properties) are based on the analysis of verbs and their arguments as defined lexically by subcategorization frames [97]. A major challenge posed by this kind of approach to relation extraction is the identification of the right level of abstraction when it comes to determining the most specific restrictions holding for domain and range [98]. This also holds for more statistical techniques based on collocations [99] or association rules [100, 101] – even though the latter anyway demand for a certain degree of user interaction as the generated relationships most often lack meaningful labels (see Kavalek and Svátek [102] for some relation labeling experiments). A semiautomatic approach to refining logically

complex domain-range restrictions of ontological properties has been developed by Rudolph and Völker [103, 104].

Disjointness

Disjointness axioms have attracted only little attention in the ontology learning community so far. On the one hand, the majority of approaches to ontology learning from text still concentrate on the generation of rather lightweight ontologies. On the other hand, it is very difficult to derive negative information from purely textual data. A classification-based approach presented in [105] therefore relies on heterogeneous types of evidence, in order to determine the disjointness of any two classes. Besides, logical methods based on inductive logic programming (e.g., [106, 107]) or formal concept analysis (e.g., [108, 109]) potentially generate disjointness axioms as a by-product of a more general learning algorithm.

Other Axioms

Even more difficult appears to be the automatic acquisition of arbitrary axioms. While initial blueprints already exist [110, 111], the quality of the learned axioms is still insufficient for most practical applications. Purely logical approaches could yield better results in terms of precision, but the sheer complexity of this task overtaxes the capabilities of many state-of-the-art concept learning approaches. Therefore, recent research and development efforts are aiming to facilitate the import of relevant axioms and modules from existing ontologies [112].

13.2.3.2 Tools and Implementations

In order to facilitate the integration of automatic or semiautomatic approaches into the overall ontology construction process, several *ontology learning tools* and *frameworks* have been developed in recent years.

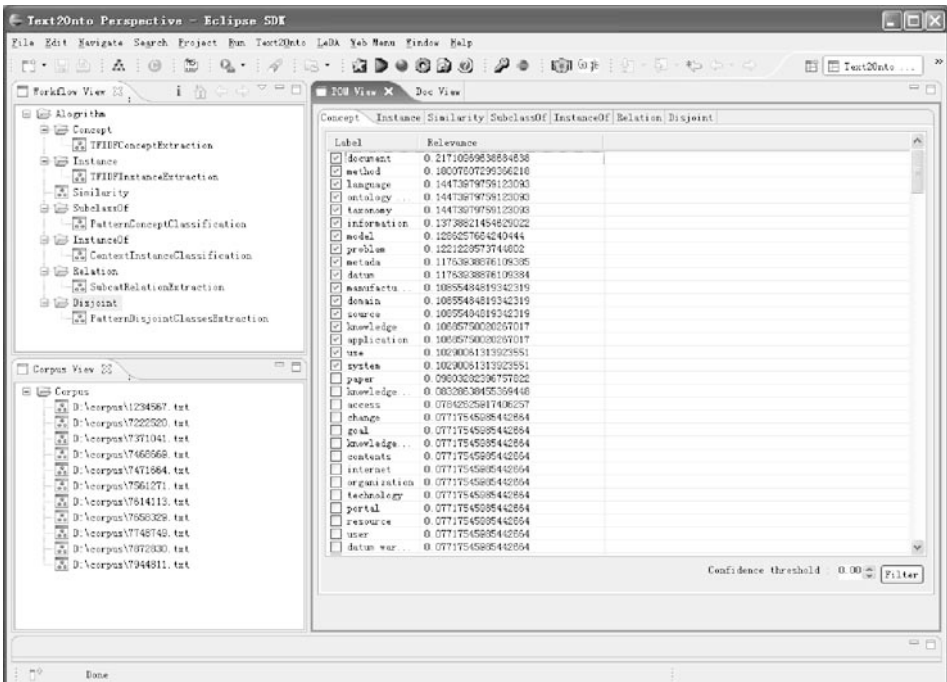
An *ontology learning framework* is a platform that (1) provides multiple ontology learning methods, each of which completes one or more ontology learning tasks; (2) guides the user in selecting, configuring, and applying these methods; (3) integrates the results of each method into a common knowledge model; and (4) supports the user in reviewing and exporting this knowledge model. These requirements are to a certain extent fulfilled by the vast majority of today's ontology learning frameworks (see ▶ [Table 13.6](#)). ▶ [Figure 13.3](#) shows a screen-shot of [117], a framework for ontology learning from text that is available as a plug-in for the NeOn Toolkit (see ▶ [Sect. 13.2.1](#)). It is composed of different views for the configuration of the ontology learning process (i.e., corpus selection and workflow composition) and the tabular presentation of the results, which can be exported into the Toolkit's editor perspective as an OWL ontology.

In addition to the above-mentioned frameworks, most of them focusing on ontology learning from text, several tools have been developed in order to support the acquisition or refinement of more complex axiomatizations, for example, by means of formal concept

■ **Table 13.6**

Ontology learning frameworks

Name	Institution	Reference
ASium	INRIA, Jouy-en-Josas	Faure and Nédellec [92]
WEB → KB	Carnegie Mellon University	Craven et al. [113]
TextToOnto	AIFB, Karlsruhe Institute of Technology	Mädche and Volz [114]
Hasti	Amir Kabir University, Teheran	Shamsfard and Barforoush [110]
OntoLT	DFKI, Saarbrücken	Buitelaar et al. [115]
DOODLE	Shizuoka University	Morita et al. [116]
Text2Onto	AIFB, Karlsruhe Institute of Technology	Cimiano and Völker [117]
OntoLearn	University of Rome	Velardi et al. [118]
OLE	Brno University of Technology	Novacek and Srmz [119]
OntoGen	Institute Jozef Stefan, Ljubljana	Fortuna et al. [120]
GALeOn	Technical University of Madrid	Manzano-Macho et al. [121]
DINO	DERI, Galway	Novacek et al. [122]
OntoLancs	Lancaster University	Gacitua et al. [123]



■ **Fig. 13.3**

Text2Onto plug-in for the NeOn Toolkit

analysis (FCA) or inductive logic programming (ILP). Some of the most well-known implementations are listed in [Table 13.7](#).

Despite the progress made in recent years especially in terms of logical or lexico-logical ontology learning, the formal quality of ontologies that can be acquired by state-of-the-art ontology learning tools or frameworks is still insufficient for many practical applications. A tighter integration with ontology engineering methodologies and automatic means for ontology evaluation could help to avoid automatically introduced modeling errors and to reduce the required amount of post-processing for ontology learning. [Sect. 13.2.4](#) takes a closer look at several past and ongoing efforts aiming to introduce best practices and theoretically well-founded quality criteria into automatic ontology acquisition.

13.2.4 Methodological Aspects of Ontology Learning

In recent years, a considerable amount of research has focused on the integration of methods for automatic ontology construction into general ontology engineering methodologies [125] as well as into specific methodologies for collaborative [126] or pattern-based [127] ontology design. At the same time, the ontology learning community has acknowledged the need for incorporating methodological guidelines and best practices into ontology acquisition approaches, for example, by applying ontology design patterns to learned ontologies [128]. Both directions of research, which could be referred to as *semiautomatic ontology engineering*, are among the most important topics for future studies on the Semantic Web. Only if one is able to combine the efficiency of automatic knowledge acquisition approaches with the skill and experience of human ontology engineers in a methodologically sound and effective way, one will eventually succeed in generating ontological resources on a large scale.

It is, however, a long way to go and the ontology learning community will have to show that it is heading in the right direction. Appropriate methodologies and benchmarks, for example, in the form of corpora or gold standard ontologies [129], for evaluating the effectiveness of ontology learning methods. A large fraction of these benchmarks should be tailored to particular applications or domains, thereby fostering the development of highly optimized ontology acquisition methods. Even though most of today's ontology

Table 13.7

Tools for logics-based ontology learning

Name	Institution	Reference
OntoComP	University of Dresden	Sertkaya et al. [124]
RELExO	AIFB, Karlsruhe Institute of Technology	Völker and Rudolph [104, 109]
DL Learner	University of Leipzig	Lehmann et al. [107]
YINYANG	University of Bari	http://www.di.uniba.it/~iannone/yinyang/

learning research still concentrates on general-purpose approaches, a constantly growing interest in the automatic acquisition of biomedical ontologies [122, 130], for example, motivates a diversification of the field. The development of specialized ontology learning and evaluation methods will also be driven by further application scenarios and opportunities, for example, in the field of product or service configuration, arising from an increasing expressiveness of learned ontologies.

The automatic or even semiautomatic generation of ontologies remains a challenging endeavor. However, there is hope that some of the guidelines and best practices derived from previous experiences in manual ontology design can find their way into automatic approaches to ontology learning and evaluation.

13.3 Example Applications

In the context of the Semantic Web, ontologies play a key role in that they provide the semantic vocabulary that semantic applications build on for the exchange and interpretation of Web data and information. Depending on the use-case scenario at hand, different features of ontologies are used in different ways within different types of applications for aiming at different goals, be it search, integration, or organization of knowledge.

This section will identify various kinds of usage of ontologies in Semantic Web information systems and relate them to typical application areas, before concrete usage examples with varying application domains will be presented.

13.3.1 Generic Functionalities of Ontologies in the Semantic Web

Regarding the wording used throughout this section, please note:

1. The term *Semantic Technologies* or *Semantic Web Technologies* shall denote the whole range of Computer Science and Artificial Intelligence methods and tools typically used and typically playing together in applications that rely on a formal ontology (or several ontologies, respectively) and on explicit, ontology-based metadata for information items or information systems – in order to enable better information search, integration, processing, or management, especially in distributed and open scenarios. Such Semantic Technologies comprise core aspects like ontology engineering and management, as well as metadata creation and management, but also contributing and underlying base technologies like natural language processing or automated reasoning.
2. Furthermore, terms like *Semantic Web applications*, *ontology-based applications*, *Semantic Technology applications*, *semantic applications*, etc., are used synonymously; that is, *Semantic Web-based* and *ontology-based* is meant interchangeably in this section, although, outside the Semantic Web scenarios, there are also other usages of formal ontologies, namely, in Multi-Agent Systems, or in Expert Systems.

3. Lastly, for the purpose of this application survey, the ideas of *Semantic Web* and *Corporate Semantic Web* are considered to be very similar. Of course, behind the firewall of a company, an Intranet application may be much easier to realize than a similar Internet application, from technical and from nontechnical points of view (trust, standards compliance, incentive systems, etc.); but, nevertheless, many company-internal information landscapes provide challenging-enough problems; some corporate Intranets today are bigger than the Internet was 10 years ago; companies must extend the scope of their electronic communication and collaboration sphere toward customers, suppliers, whole value creation networks; and the dynamics that many businesses have to cope with is really impressive; so altogether, Corporate Semantic Web is a challenging area, which may provide some clearer business cases than the whole Web can offer.

Before presenting *concrete* applications of ontologies in the Semantic Web, some *generic functionalities* of ontologies will be derived from definitional elements of ontologies as introduced above:

- *Formality* – This is the basis for a high achievable degree of automation when processing ontological knowledge or ontology-based metadata (and, as an effect, a potentially high level of automated “intelligence”). Obviously, the more heavyweight an ontology is (higher degree of formality), the more and more powerful automated and intelligent processing services are possible. Of course, a high degree of formality with its potential benefits must be traded in many scenarios against design and maintenance costs, and may be unnecessary or even inappropriate in others (e.g., in rather informally structured knowledge-organization tasks where the final knowledge consumer is a human, navigating user).
- *Explicitness* – Making modeling decisions explicit to a large extent, formally modeling them, or even modeling *redundant* aspects in an expressive language can be the basis for better human understanding where and how to use (or reuse) an ontology or ontology-based metadata, but it may also be the basis for the (semi)automatic mapping of different ontologies or for more automated inferences.
- *Consensus* – Consensus on modeling decisions is the most effective way to achieve data and system interoperability. So, harmonizing heterogeneous structures and viewpoints can be facilitated by a broad sharing of agreed-upon knowledge – in the extreme case, this leads to standardization. Of course, standardization is not always possible or (from a political, economic, or psychological point of view) achievable/desirable. But even where no consensus about specific domain- or task-related ontological knowledge can be achieved, the consensus on at least the basic knowledge representation formalisms (Semantic Web standards) makes interoperability already easier. Furthermore, formality and explicitness support the process of finding consensus as they allow one to communicate clearly what different parties are talking about. Lastly, an increasing number of publicly available ontologies will increase the degree of model reuse, which *implicitly* fosters consensus through de facto standardization.

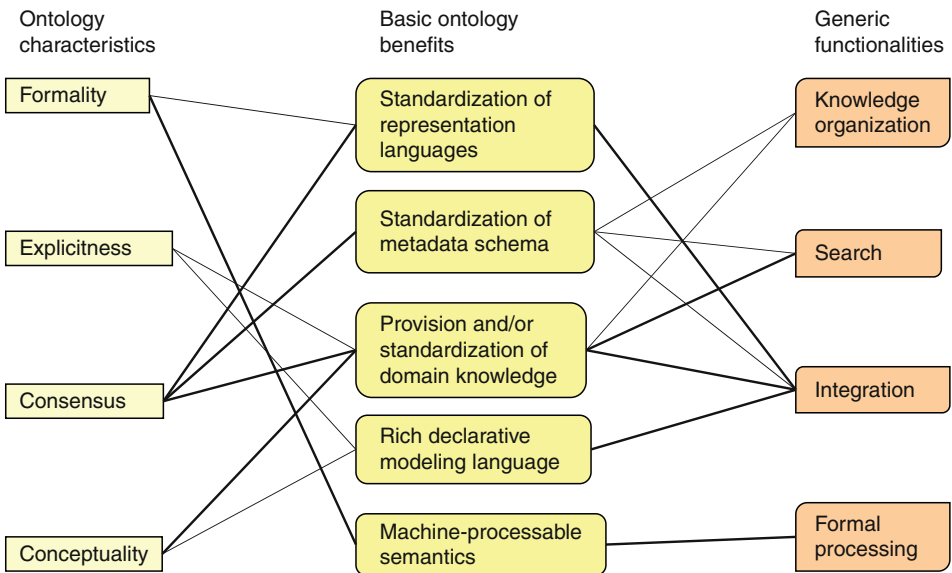
- *Conceptuality* – This way of declarative modeling a domain’s knowledge structures is the basis for all search-support functionalities offered to humans, for example, for navigation or for query disambiguation or reformulation (see below).

These characteristics contribute, to a different extent, to a number of potential basic benefits, which ontologies can offer to applications, namely (see [Fig. 13.4](#)): (1) *standardization of representation languages* (like RDF, RDFS, OWL); (2) *standardization of metadata schema* (like WSMO for Web Services); (3) *provision and/or standardization of domain-knowledge* structures; (4) *rich declarative modeling* languages; (5) *machine-processable semantics*, including automated reasoning facilities.

In addition to these basic benefits, which can be gained from using ontologies in general, for the specific case of *Semantic Web* ontologies, two further advantages can be mentioned, namely (6) that Semantic Web ontologies aim at being *incrementally extensible*, and (7) that Semantic Web ontologies and knowledge bases are designed for being built, extended, and maintained in a *distributed* manner.

The above listed basic benefits lead to a number of *generic functionalities* realizable through ontologies in (distributed, Web-based) Information Systems:

- *Knowledge and Information Organization* – Many knowledge and information management systems rely on informal, only partially structured, and hardly machine-processable representations of knowledge, such as books, personal notes, drawings, presentations, e-mails, or any kind of multimedia documents. These can be stored in digital libraries, document management systems, personal file systems, etc. Often, such storage systems are organized with the help of metadata that describe the stored



■ Fig. 13.4

Ontology characteristics lead to basic benefits and generic functionalities

knowledge items. To express such metadata, some systems employ a metadata schema in the form of an *information ontology*; moreover, content descriptions of knowledge items in metadata can refer to *domain ontologies*.

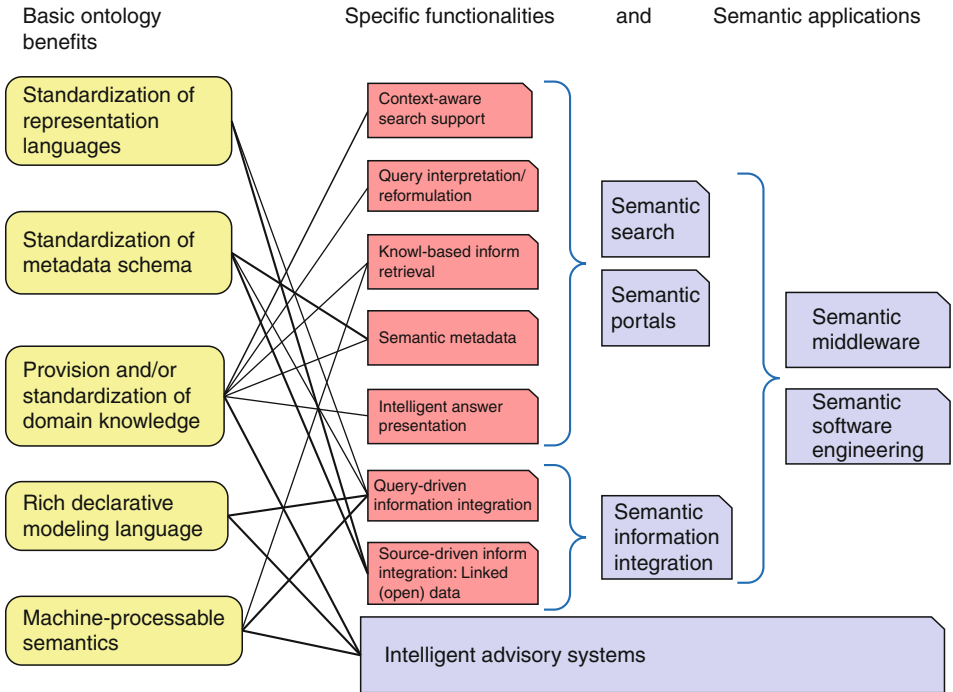
- **Search** – Almost all Semantic Web use cases deal, in one way or the other, with searching for Web-accessible resources. Ontologies can support and improve practically all aspects of information search (see [▶ Sect. 13.3.2](#)).
- **Integration** – If data and information from more than one source shall be processed together, different data schemas must be linked together or (partially) be mapped onto each other. At least three cases can be considered: (a) If the data schemas to be mapped are expressed as ontologies, (*semi*)*automatic* mapping is facilitated through ontologies because rich and explicit, possibly redundant, models in a language with an unequivocal formal semantics provide a good starting point for that. (b) Even in the case of fully *manual* schema mapping: If schemas are expressed as ontologies (which typically come together with rule languages like SWRL or F-Logic), the rule-based formulation of schema mappings is an easy-to-use, yet powerful mechanism. (c) Independent from the issue of schema mapping *at query time*: There are also scenarios where it is wished to link data from different sources already at the information-provider side. This is facilitated much by Semantic Web standards, as they are designed for that purpose.
- **Formal Knowledge Processing** – Automated reasoning over ontology-based metadata and/ or ontological background knowledge is a functionality that may be used in either of the three other functionalities above, be it to achieve a higher degree of automation, to provide new functionalities enabled through executable knowledge representations, to deduce implicit knowledge, or to validate and verify knowledge formulated by the user.

These generic functionalities are jointly employed and instantiated in manifold ways in the more specific functionalities and classes of semantic applications collected in [▶ Fig. 13.5](#), which are further discussed below. Concrete application examples often combine aspects of more than one of these categories.

13.3.2 Applications of Ontologies in the Semantic Web

13.3.2.1 Semantic Search

Semantic search in Intranets and in the Internet is a very widespread application of ontologies which can be characterized by the following working definition: *Semantic search employs semantic technologies to support human or automated agents in the process of finding – in a given search context – from one or more (Web-based) information sources those information item(s) most appropriate for satisfying a given information need. Major aspects of this task are often (1) that the search system obtains a detailed and unequivocal interpretation of the semantic intent of the information need at hand; and (2) that the search system employs semantic metadata and/or domain-specific background knowledge about available information sources and information items for precisely and comprehensively answering this information*



■ Fig. 13.5

Ontology benefits support specific functionalities and semantic applications

need. Often, some connected activities are also included, especially for helping the searching agent with making the best use of the found information, for example, by specific answer-set post-processing or result presentation.

A wide variety of approaches is subsumed here. The schematic illustration of an information-search process in ▶ Fig. 13.6 shows points of application for ontology-based support:

1. *Context-aware search support*: The process starts with an agent being aware of a context-dependent information need; the agent articulates this information need in a form he or she is able to use. This form may be a list of search terms, a question in natural language, a document, which might be similar to the expected answer document, a SPARQL query, etc. Especially human agents who are not always aware that they have an actual information need can be supported by context-aware information systems: They detect the information need and can notify the user or even pose the query automatically to the search system – thus proactively delivering potentially relevant information [131, 132]. In all cases, *ontology-based personalization* may support information-need articulation with knowledge about the user's prior knowledge, major interests, specific wording, specific search constraints, etc. *Summary*: Here, ontologies contain background knowledge and offer KR formalisms for user modeling, context modeling, activity rules, query-adaptation rules, etc.

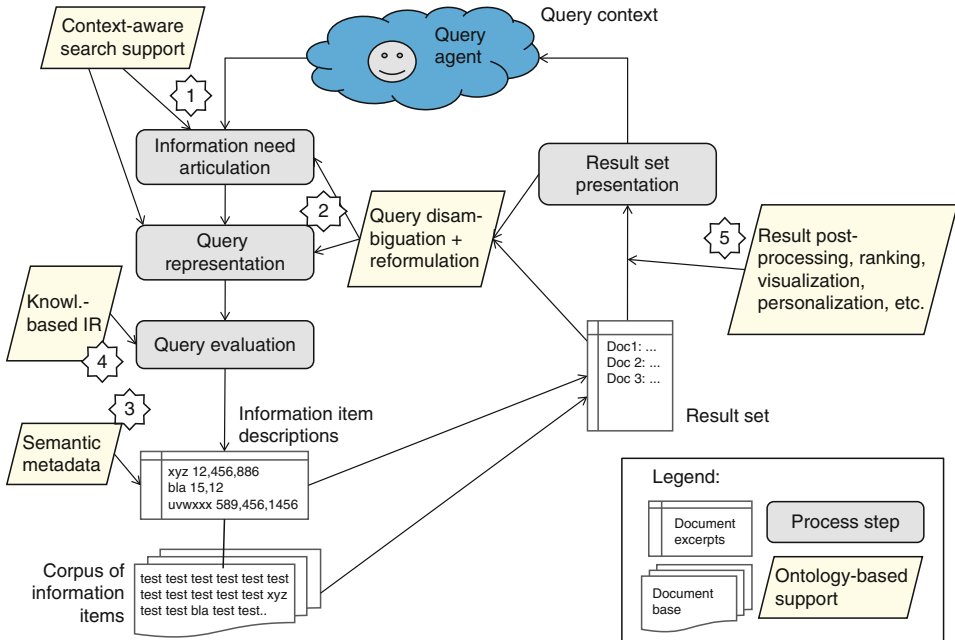


Fig. 13.6

Semantic search process

- Query disambiguation and reformulation*: If the articulated information need (for instance, a natural language question) and the query representation for the query evaluation back-end, for example, a Boolean query over document metadata) differ in their syntax, a translation process must take place. Even in the case of the same syntax for human query formulation and query evaluation, it may be derived from the ontological structures, from context knowledge or from analyzing the available information sources, that the formulated information need does not match the conceptual structures of information-item descriptions, or that it is ambiguous, too vague, too specific, or even inconsistent; then it must be refined or reformulated (see also [133]). In this disambiguation and semantic interpretation process [134], ontologies can (A) help to overcome natural language ambiguities through (A1) encoded knowledge about the relationship between natural language expressions and formal concepts and (A2) encoded content-specific background knowledge; they can (B) provide a fine-grained vocabulary for the formulation of very precise and unambiguous as well as very complex questions (e.g., using attributes and relations). In Fig. 13.6, the arrows from phase (5) back to the disambiguation and reformulation step indicate that also too many, too few, or bad answers can give reason to change the query posed to the system. This may happen interactively with the user, or (heuristically) automated with query-reformulation rules. In Information Retrieval (IR), such approaches are well-known as *query expansion*, whereas here, the term *query reformulation* is preferred because the query may be extended, refined, or completely

changed. If the knowledge space exploited for query formulation is given through a domain ontology, one would talk about *concept-based query reformulation* [135], in contrast to a *lexicon-based* query expansion, which only considers *wording aspects*. *Summary: Altogether, the domain ontology provides the knowledge space within which the query can be varied.*

3. *Knowledge-Based Information Retrieval*: At the heart of each search solution stands the query evaluation subsystem which matches the formal representation of an information need against the formal representations of information-item content in order to identify those information items (text or multimedia documents, database records, etc.) with the highest probability of satisfying the information need at hand. For doing that, several IR paradigms have been devised, the *Vector Space model* [136] working on textual keywords being the most prominent one. When information items are represented by declarative metadata records, the keyword-based paradigm can be replaced or extended in the most simple case by the *Boolean retrieval model* [136]. If semantic metadata are fully expressed in or do at least refer to logics-based representations in a Semantic Web language, one can use a *logic-based IR approach* [137]: Here, the relevance of an information item for a given query is logically inferred on the basis of given metadata facts and ontological background knowledge (by deductive or abductive inference; often also taking into account the uncertainty and vagueness of IR probabilistic, possibilistic, or fuzzy logic; for systems based on description logics, there are also approaches implementing retrieval as *classification* inference [138]). Concrete semantic search implementations often choose a hybrid approach combining text-based and metadata-based retrieval. Another form of knowledge-based IR, which is not logics-based, but can exploit ontological background knowledge, is *similarity-based retrieval* coming from Case-Based Reasoning [139]: Here, both the information items and the query are represented as sets of attribute-value pairs; the expected utility of an information item for a given query is assessed through a weighted sum of local value-similarities per attribute, compared between the stored item and the query. Ontologies can here define the ranges for individual attribute values and then provide the basis for local similarity assessment. *Summary: Here, ontologies provide the knowledge base for assessing relevance of information items, their formal semantics is the starting point for declarative retrieval models, and they can provide background knowledge for nontrivial inferences bridging larger conceptual distances between query concepts and concepts in the descriptions of information items.*
4. *Semantic metadata*: Describing information items or underlying information sources through semantic metadata means to make sets of statements that instantiate an information ontology (aka metadata schema); these metadata statements may refer to one or more domain ontologies for describing the content topics an information item is talking about. Mechanisms for including links to domain ontologies (or other rich knowledge models) for sophisticated content description are already foreseen in many metadata standards like the Dublin Core for general electronic documents, or the IEEE Learning Object Metadata standard (LOM) for eLearning resources. Besides the possibility of giving expressive content descriptions, rich metadata can also bring

further benefits: (a) they allow one to describe real metadata, that is, aspects of an information item, which are not *contained* in this item, for example, the creation context of a lesson learned from a project, the expected prior knowledge for an eLearning lesson, etc.; (b) they allow one to refer to finer-grained pieces of knowledge than the whole information item; for example, they could represent individual chapters of a book; (c) vice versa, they can also represent more coarse-grained content objects, for example, a whole document collection or a set of complementary, interlinked eLearning resources (motivational examples, a mathematical theorem, its corollaries, example applications, and related exercises); (d) declarative metadata allow one to represent non-text information items, like multimedia content; (e) through standardized, homogeneous metadata, information items from different heterogeneous sources can be brought together, put into the same context, and interlinked. *Summary: Regarding semantic metadata, Semantic Web ontologies provide rich and machine-processable representation languages, they can standardize metadata schemas and provide standardized domain knowledge for content description.*

5. *Post-processing of search results:* If a number of potential answers have been retrieved, several kinds of post-processing may happen, which can be supported by ontologies: (a) results can be ordered and *ranked* according to declarative rules; (b) when retrieving *informal* documents (in contrast to *factual* knowledge), information extraction algorithms can be applied for gaining factual knowledge from the documents; this knowledge can be stored in ontological data structures and be *further processed* with knowledge-based methods; (c) especially in the case of large answer sets or complex information spaces, some aspects of the answer set are *visualized* for further human inspection and browsing; for instance, documents can be arranged according to their relevance to domain topics or visualized with their interrelationships as semantic networks; (d) also the answer post-processing and presentation can be subject to personalization and context-specific adaptation. *Summary: In result post-processing, ontologies provide the data structures for formal knowledge representation, the specific post-processing knowledge, or the domain-knowledge structures as the backbone for visualization, respectively.*

Please note two simplifications of the schematic semantic search process:

1. Only *one* information source is considered; in general, many sources may be available. Then, ontology-based metadata may be used to select the most appropriate source(s) for a given query. It can happen that the query evaluation subsystem must then dispatch several partial queries to the respective information sources, possibly in different query languages, and must reintegrate the results from different sources – which essentially amounts to the use case of query-based information integration (see [Sect. 13.3.2](#)).
2. Only *query-based* information access is discussed. But especially when human agents drive the scenario and when a relatively unknown information domain is to be explored, a *navigational* access may be more suitable. Combinations of both are also possible.

A general remark: In today's still standard case of full-text indexes for information-item representation (i.e., when neglecting all possibilities of ontology-based metadata), the query

evaluation would typically rely on Vector-based IR methods. Also with such a fully conventional search engine in the back-end, the above explained approaches (1), (2), and (5) will perfectly work, exploiting the ontological knowledge to enhance the user interface, without a need of changing the legacy systems. Just the query evaluation (3) itself can only be radically improved if the information-item descriptions are based on semantic metadata (4).

13.3.2.2 Semantic Portals

A Web Portal is a unique place in the Internet or a corporate Intranet to gather and present information from diverse sources in a unified manner; typically collecting and syndicating content (streams) about one specific topic, domain, region, or company, facilitating the work of one topic-oriented community (community portal), or the collaborative effort of a team with a dedicated task (project portal). Content types may include news and up-to-date information streams, e-mail, as well as (multimedia) documents. Web portals often provide a consistent look-and-feel for heterogeneous input, with access control and interfaces for multiple applications, for example, information push services (like RSS feeds) or comfortable information access with mobile devices. Based on [140], the following typical functionalities can be listed:

- *Information Supply* – Users have easy means to submit information and make contributions to their community(ies).
- *Information Management* – Portal administrators can easily integrate new (static or dynamic) information sources, keep the content consistent, change layouts, etc. This can include the (automatic) establishment of *links* between content items, which are not existing at the level of the individual content items.
- *Information Browsing* – Domain-knowledge structures are the basis for navigation menus, faceted browsing, information visualization, etc.
- *Information Search* – Unified search over heterogeneous content is provided.
- *Personalization* – Individual configurations for layout, information-delivery modalities, content selection, etc.

Compared to a conventional Web Portal, a *Semantic Portal* can be characterized by: (1) a domain ontology used as the central, harmonized topic structure for knowledge organization, navigation, and visualization; (2) semantic search mechanisms; and (3) Semantic Web languages for internal data management – which facilitates declarative approaches for further functionalities like personalization [141] or consistency checking of content. **OntoWeaver** [140], **Ontoviews** [142], or **SEAL** [143] are well-known Semantic Portal frameworks; see [144] for an overview.

Below, some examples for semantic search and semantic portal solutions are listed:

- References [145–147] analyzed and compared numerous academic and commercial semantic search tools with respect to some of the functionalities explained above (see also http://swuiwiki.webscience.org/index.php/Semantic_Search_Survey).

- There are many commercial tools for *semantic enterprise search* implementing some of the aspects discussed. For instance, ontoprise's Semantic Miner realizes many of the aspects under (2), (3), (4), and has a number of operational installations in large enterprises (see <http://www.ontoprise.de/>).
- Reference [148] presents a *corporate-search solution* for large, heterogeneous document collections dealt with by engineers in Rolls-Royce plc. Metadata can be manually edited, semiautomatically created, or extracted from legacy documents in a fully automatic manner. All document metadata are stored in OWL and RDF triples. The retrieval approach is hybrid and combines text-based with metadata search.
- Some examples for *similarity-based retrieval* can be found in the area of competence and skill management when human skill profiles are compared with, for example, job or actual problem descriptions, for instance, in Expert Finder systems [149].
- Many projects build portals for *cultural heritage* information (like descriptions of museum exhibits, content of national archives, or libraries [150, 151]). Besides issues like faceted search or metadata interoperability, these scenarios typically involve time and spatial aspects in metadata representation and querying – which is a special challenge for ontology-based approaches. References [152, 153] describe large-scale demonstrators for aspects such as result clustering and semantic recommendation (see also ► [Multimedia, Broadcasting, and eCulture](#)).
- Numerous authors discuss semantic annotation of *biomedical* resources and subsequent search and question answering [154, 155]. **HealthFinland** is a semantic portal for health information in Finland where a Finnish ontology for laymen searching health information is constructed from several input ontology sources such as the Medical Subject Headings (MeSH) in SKOS format [156]; the content and metadata-creation workflow and tool support are presented as well; finally, the user services and interface design as well as their evaluation are discussed.
- In the *AgentDysl* project [157], retrieved eLearning resources for an electronic training system for dyslexic learners, are at real time arranged and adapted such that they best support the individual learning style and the current mood and psychological status of the learner, thus implementing a rule-based, context-specific result post-processing.

13.3.2.3 Semantic Information Integration

Data or information integration as the basis for data reuse, for query answering from multiple (heterogeneous) sources and for interoperable software systems in organization-internal (Enterprise Application Integration, EAI) or cross-organizational settings (eBusiness, Business-to-Business Communication, B2B) is a long-standing goal of Computer Science and a major motivation for the interpretation of the Semantic Web as a *Web of Data*. Jointly exploiting structured or semi-structured information from multiple sources has been researched in the Database community, under labels such as federated information systems, federated databases, etc. In principle, there are two major approaches: (a) the traditional way of looking at the topic, where data or systems,

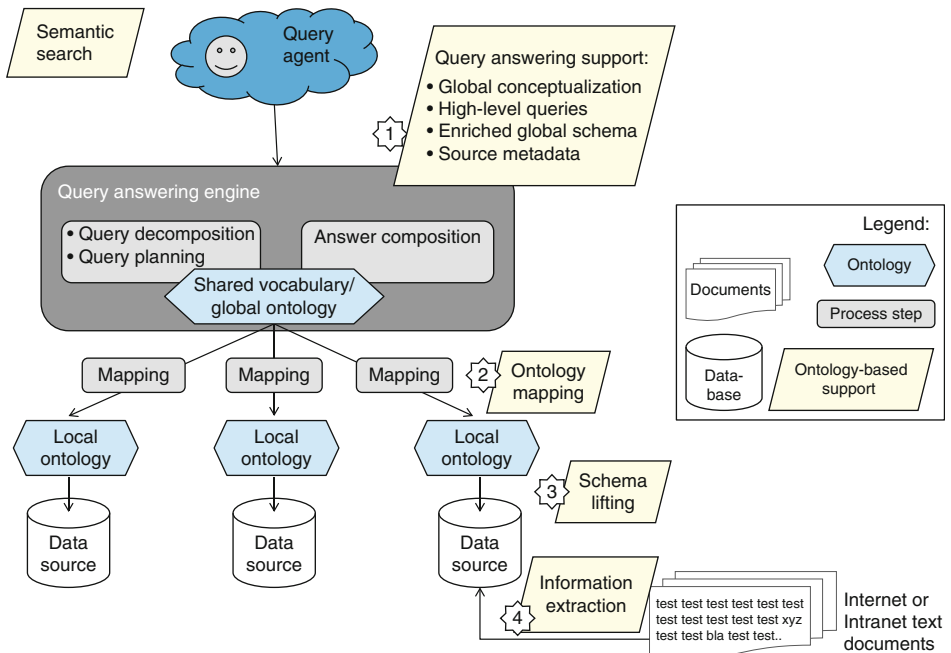
which were not designed for interoperability shall be queried in one interface and information pieces from different existing information sources must be found, combined, and aggregated – this is called *Query-driven Information Integration*; (b) the relatively novel approach of *Linked (Open) Data* where people intentionally use Semantic Web standards in order to foster the reuse and linkage of their data in manifold application contexts. This is a bottom-up approach, that is, *data-source driven*.

Query-Driven Information Integration

If data from different systems shall be integrated, a number of different kinds of heterogeneity can occur, which must be dealt with ([158] list some dimensions of heterogeneity). To this end, several integration approaches and architectures are possible; in the Semantic Web area, complying with Wiederhold's wrapper-mediator-architecture [159], most widespread is (according to the terminology in [160]) the *hybrid ontology approach*: A local ontology is built for each data source, representing its database schema in a standards-compliant, knowledge-rich manner. There is one global vocabulary or ontology, which is used at the query side where all local ontologies are mapped onto.

► *Figure 13.7* illustrates the general idea, as well as points of application for ontology support in data integration, based on [161]:

1. *Query-answering support*: A shared central vocabulary provides a *comprehensive and unifying conceptual view* on the application domain, independent from partial and



■ **Fig. 13.7**

Ontologies in query-driven information integration

heterogeneous, maybe implementation-driven, local schemas, or ontologies at the level of the individual information sources. This allows one to formulate *high-level queries* without knowledge about the different back-end data sources. Indeed, it is not even necessary to know which data sources exist, because the user can employ the global ontology as the only reference point for queries. It is also possible that the query answering engine uses metadata about data sources for *query decomposition*, *query planning*, etc. Such metadata may contain information about the content coverage of certain data sources, about data-source availability, about quality of content, or about costs associated with data access. Furthermore, *answer composition* from individual partial query results can be carried out declaratively, for example, in a rule-based manner. Lastly, at the level of the global ontology, *model enrichment* can happen, that is, that additional knowledge (in database terms, e.g., by a view mechanism) is inferred from data from different sources such that the global access layer may contain facts, concepts, and relationships, which do not exist in any *single* data source, but only virtually when looking simultaneously at the *whole* data landscape. A problem of answer composition from multiple sources is called *data matching* [162]: Identify that information from different sources refers to the same real-world object. Here, an ontology can contribute background knowledge and can be the hook for sophisticated reasoning about identity.

2. *Ontology mapping*: If both local data schemas and global schema are represented as ontologies, *declarative schema mediation* is possible: Mappings can be found easier because all schemas are represented in one standardized language; mappings can be represented in a rule-based manner – which is easier to formulate and maintain than procedural representations; and the manual formulation of mappings is often supported by visual editors. Using rich ontological knowledge, the *finding of mappings can be partially automated*. Creating ontology mappings is discussed in a large research community and has produced many valuable results [163, 164].
3. *Schema lifting*: As said above (under 2), the first step for ontology-based information integration is to express all local data schemas in the same ontology language. This step – in particular, lifting the data schema of a relational or an XML database – can also be (partially) automated by Semantic Web technologies (see, e.g., [161]).
4. *Information extraction*: Up to now, only discussed is the case that the data sources to be integrated are (semi-)structured databases like relational or XML databases. But, in general, all considerations still hold true if the back-end (partly) consists of less structured, informal information sources (like HTML pages with free text); these can be analyzed with NLP methods and – to some extent – be formalized, typically, by *information extraction* (IE) techniques, which identify specific kinds of information in texts and create structured representations for them. The result of such extraction tools can then either be stored persistently in a database or be created ad hoc at query time and queried like a structured data source (see also [Semantic Annotations and Retrieval: Manual, Semiautomatic, and Automatic Generation](#)). If such IE techniques are employed, the results may be directly stored as ontological assertions. An example is the MUSING system [165] which uses ontology-based IE to fill an integrated knowledge base from distributed, unstructured Web resources.

More information on the status of semantic information integration can be found here:

- Reference [166] is a comprehensive survey of semantic integration approaches, including the classification of many systems. Reference [162] gives a survey of semantic integration approaches seen from the database community, including a list of open problems.
- Reference [167] shows with many examples how to do rule-based information integration with F-Logic in the automotive area for *company-internal* purposes, whereas [168, 169] discuss *cross-company* information exchange in the automotive retail domain, based on their suggested **STAR** RDFS reference ontology for automotive retail.
- The **NeuroBase** project [170] is an example application for integrating heterogeneous information from different experimental sites, hospitals, and research centers in the area of cognitive neurosciences. It implements a standard wrapper-mediator architecture with a central unifying ontology.
- References [171, 172] present the **Crossvision Information Integrator**TM product of Software AG, which almost completely supports the architecture in [▶ Fig. 13.7](#).

Source-Driven Integration: Linked (Open) Data

The Linked (Open) Data initiative (LOD) ([173], see also [▶ Semantic Annotation and Retrieval: Web of Data](#) of this handbook) is a relatively new endeavor, which is based on a few best practices and technological principles for publishing and connecting structured data on the Web, extended by machine-readable metadata. The technical realization is based on the thorough use of URIs, the compliance with open Web standards like RDF and SPARQL, and the extensive use of links between data. Already in the recent few years, a global data space evolved containing 4.7 billion RDF triples interlinked by 142 million RDF links (figures from May 2009) – the “Web of Data” – connecting data from diverse domains such as people, companies, books, scientific publications, films, music, television and radio programs, genes, proteins, drugs and clinical trials, online communities, statistical and scientific data, and reviews (see [▶ Fig. 13.8](#) from <http://richard.cyganiak.de/> for an overview of data sets included until July 2009). Applications comprise, for instance, *Linked Data browsers*, which allow users to start browsing in one data source and then navigate along links into related data sources; or *Linked Data search engines* that crawl the Web of Data by following links between data sources and provide expressive query capabilities over aggregated data, similar to how a local database is queried today. Reference [173] enumerates a number of *Linked Data publishing* tools for both content in RDF stores and in non-RDF legacy databases. SparqPlug [174] is a service that enables the extraction of linked data from legacy HTML documents on the Web that do not contain RDF data. The service serializes the HTML DOM as RDF and allows users to define SPARQL queries that transform elements of this into an RDF graph.

A prominent example for an LOD application is DBpedia Mobile [175], a location-aware LOD browser running on a mobile device like the iPhone. DBpedia Mobile supports a tourist exploring a city. Based on the current GPS position of the mobile device, DBpedia

Mobile provides a location-centric mash-up of nearby locations from DBpedia, plus associated reviews from Revyu, and related photos accessed through a linked data wrapper around the Flickr photo-sharing API. DBpedia Mobile also enables users to publish their own current location, pictures, and reviews as Linked Web Data. Published content is not only described through geographic coordinates, but is also interlinked with a nearby DBpedia resource.

The LOD approach thus relies on a *pay-as-you-go data integration* approach [176] based on a mixture of using common vocabularies together with data source-specific terms that are connected by mappings as considered necessary.

Regarding the role of ontologies for LOD, simple information and domain ontologies like FOAF, SIOC, SKOS, DOAP, vCard, Dublin Core, OAI-ORE, or GoodRelations are used wherever possible. If new terminology is defined, it should be made self-describing by making the URIs that identify terms Web dereferencable. This allows clients to retrieve RDF Schema or OWL definitions of the terms as well as term mappings to other vocabularies. Linked data should be published alongside several types of metadata, in order to increase its utility for data consumers and quality, including provenance meta-information about data-creation time and procedures.

A remark on the practical relevance of the LOD initiative: One might wonder whether the idea of opening own databases for the public is naive or altruistic and what should be the economic incentives for doing so in business. There is a really huge noncommercial sector of information producers, like public authorities in eGovernment and eParticipation (this includes, e.g., the area of environmental information services or statistics offices), the whole scientific area, the area of NGOs (nongovernmental organizations) with altruistic motivations like charity that often have a strong interest in getting their information public, and finally the large area of information without active enforcement of copyrights, for example, historic documents. Moreover, even in the business sector, there are sometimes noncompetitive areas (e.g., research collaborations between companies, which search their Unique Selling Propositions in the later commercialization of research results).

Moreover, the LOD methods and principles can, of course, also be applied *within* an enterprise. For instance, the British Broadcasting Corporation (BBC) uses linked data internally as a lightweight data integration technology [177]. Formerly, BBC's numerous stations and channels used separate content management systems. Recently, BBC started to use linked data technologies together with DBpedia and MusicBrainz as controlled vocabularies to connect content about the same topic residing in different repositories. This content is augmented with additional data from the LOD cloud.

13.3.2.4 Intelligent Advisory Systems

An Expert System (XPS) [178], or more general, a Knowledge-Based System, is a computer program that simulates the judgment and behavior of one or more human

experts who have expert knowledge and experience in a particular field. Typical problems addressed are diagnosis, configuration, planning, process monitoring, teaching, design, data analysis, or forecasting; application domains comprise medicine, mechanical, electrical, and civil engineering, manufacturing, chemistry, biology, geology, law, and many more. Typically, such a system contains a declarative knowledge base containing accumulated experience and a set of rules for applying the knowledge base. From a purely technological point of view, XPS have been mature since the late 1980s, to achieve a problem-solving performance comparable to a human's solution in quality, in many sophisticated cases. Nevertheless, XPS never had the big breakthrough expected at the time. The reasons for this include – besides exaggerated expectations and promises, and also technological problems, which can be considered solved in the meanwhile (performance, integration) – (a) that from an economic point of view, the creation and maintenance of XPS were far more expensive and complicated than acceptable in many operational settings, and (b) that it is always psychologically difficult to completely delegate complex decisions in difficult situations to a machine. So, modern applications of XPS technology often realize the idea of *intelligent advisory systems which do not fully automate a complex decision, but instead help a human user to find a decision, by, for example, (1) delivering important information (aka intelligent information retrieval), (2) helping to analyze complex or voluminous data and information streams, (3) checking complex constraints (aka automated critiquing component), or (4) suggesting partial problem solutions to the user for further human inspection and processing.*

Aspects (1) and (2) above refer mainly to semantic search and semantic information integration, whereas aspects (3) and (4) can be well implemented using “traditional” XPS techniques (which are nowadays increasingly amalgamated with Semantic Web languages and technologies). Settling upon state-of-the-art Semantic Web approaches facilitates systems interoperability and promises more cost-efficient systems engineering through the reuse of existing ontologies. Examples for the practical usage of intelligent advisory systems are listed below:

- The **Hospital Care Watch** [179] is a patient-management assistant prototype using an ontology about hospital care concepts (including hospital activities, procedures, and policies, and insurance policies, as well as medical knowledge per se) and a set of rules – for tracking the implications of medical decisions taken by physicians and other medical professionals within the context of guidelines and regulations, in order to avoid errors.
- References [180, 181] explore the combinations of ontologies and fuzzy inferences in medical application areas, such as respiratory waveform classification or diabetic food recommendation.
- The **IASO system** [182] is a mobile application which provides in a context-specific manner knowledge about a patient to a doctor outside the hospital. The system is based on three OWL ontologies formalizing, respectively, (1) patient histories in the hospital information system, (2) the description of the current patient situation, and (3) the context-specific relevance connecting elements/subsets of (1) and (2). The system is implemented in Java with the OWL API and employs the Pellet reasoner.

- Reference [183] describes an application of the ontology- and rule-based **Semantic Guide** product for supporting the customer service at a manufacturer of industrial robots. The implementation is based on ontoprise GmbH's **Ontobroker** F-Logic reasoning engine.

13.3.2.5 Semantic Middleware

In the recent decade, computer software and system architectures are increasingly moving away from monolithic, “one-program-on-one-computer” approaches toward distributed computing – putting together partial contributions from different, loosely coupled subsystems, often connected and communicating through standard Internet protocols. Manifold concrete realizations have been developed, from a virtualization of processing or storage system (as realized by *Grid or Cloud Computing*), to the composition of complex workflows from simpler, Web-accessible services (as realized by *Web Services and Service-Oriented Architectures*), or even to giving up the traditional notion of control in a program such that system behavior is emerging from the dynamic communication between partly autonomous units (as realized in *Multi-Agent or Peer-to-Peer Systems*). In all these paradigms, end-user interface and end-user application are decoupled to a different extent from concrete machines and operating systems, sometimes also from more abstract computing resources. So, all technical implementations of the aforementioned paradigms can be considered *middleware* [184, 185] components. Practically, all these middleware approaches face (to a differing extent) the following challenges at runtime:

1. *Find* the most appropriate available subsystem (Web service, peer, Grid resource, etc.)
2. Establish close *communication* between loosely connected elements (message exchange)
3. Realize an overall system *control* without a strong control paradigm

Obviously, challenge (1) might be improved by *semantic search*, (2) by *semantic integration* and mediation, and (3) by *reasoning* or at least some form of rule-based, declarative programming inside or outside the individual subsystem. Consequently, all paradigms have already been combined with semantic technologies:

- *Semantic Web Services* – Certainly the most active area to be discussed here [17, 186, 187]. A number of competing approaches for knowledge-rich Web service metadata have been suggested (see above: Service Description Ontologies) and been proposed for standardization. This falls under the above-introduced application class of *semantic metadata* in the area of semantic search. And, indeed, also manifold *knowledge-based IR techniques* for *service discovery* have been developed, comprising structural ontology matching [188], deductive retrieval using F-Logic inference for WSMO services [189], description logic inferences based on OWL-DL for SAWSDL services [190] or hybrid approaches (e.g., [191, 192] combine ontology-based type matching, logical constraint matching, and syntactic matching for WSMO and OWL-S service

profiles, respectively, in a mixed similarity measure). The benefits from using ontologies lie in the formality and expressiveness of the representation language, and in the standardization of domain knowledge, for example, about products and services. For the task of *Web service composition* [193] to combine simple functionalities for achieving more complex ones, including the adaptation of interfaces between services, which might not perfectly fit together, many ontology-based solutions have been devised, too. These sometimes employ well-known *ontology mapping* techniques, and often are just declarative approaches, which rely on the fact that Semantic Web Services are described in a high-level, rich manner on the basis of a formal language. For instance, [194] describes an interactive service composition tool for WSMO service profiles, [195] uses linear-logic theorem proving for processing DAML-S service profiles, [196] applies abductive reasoning with constraint relaxation, based on former work in AI planning [197], and [198] combines description logic reasoning with situation calculus from AI planning. Please note: Techniques developed for Semantic Web Service Management can more or less directly be reused for the organization-internal management of workflows or other activity descriptions. For instance, [199] describes an interactive workflow-composition tool for configuring scientific workflows in seismic hazard analysis (earthquake research) based on similar principles as described above; [200] model biological processes with WSML and are then able to do sophisticated pathway analyses (explorative, verification) of biological process chains. See [▶ Semantic Web Services](#) for more details on this area.


- *Semantic Grid* – Grid computing is a form of distributed computing based on a “virtual super computer,” which is composed from a huge cluster of loosely coupled standard computers [201]. These cluster computers may be heterogeneous and may be geographically distributed. It has its major applications in science, especially in the typical application fields of high-performance computing, such as simulation in meteorology, in pharmacy, geology, etc., but also in some commercial fields, for example, in the automotive or in the finance area. The vision of the Semantic Grid [202] semantically represents Grid-resource metadata with RDF such that, in principle, pretty much the same benefits are possible as described above for Semantic Web Services.
- *Semantic Peer-to-Peer* – In a peer-to-peer (P2P) distributed network architecture [203], participating computers make a part of their resources (storage space, network band-width, processing power) *directly* (without intermediary hosts or servers) available to other peers. So, each peer is both a consumer and supplier of resources. Obviously, indexing and resource discovery are challenges in large P2P networks if one wants to find the most appropriate peers for a certain purpose and avoid huge communication traffic in the network. P2P systems often implement an application layer overlay network on top of the physical network topology, used for indexing and peer discovery. Reference [204] introduced the term *semantic overlay network* for a logical peer organization which is based on content-oriented classification hierarchies. Reference [205] presents a schema-based peer-to-peer overlay network that facilitates efficient lookup for RDF-based information in dynamic environments, which uses, among others, the semantic clustering of peers based on RDFS ontologies as one routing

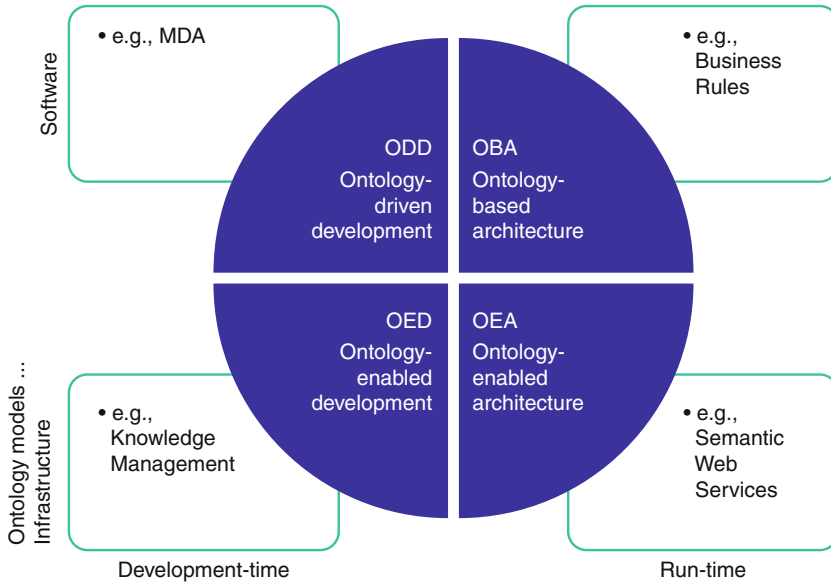
optimization technique. The approach is applied in the scenario of building context-aware applications in smart spaces [206]. GridVine [207] may have been the first semantic overlay P2P system with an explicit focus on high scalability as well as semantic interoperability through schema reconciliation; the system is based on local RDFS schemas and uses OWL to encode schema mappings. Bibster [208] was an early application of semantic P2P overlay systems in the area of sharing bibliography entries; it uses lightweight RDFS versions of SWRC as an information/domain ontology and the ACM Topic Hierarchy as a domain ontology/vocabulary; Sesame is used as a triple store.

- *Semantic Cloud* – Cloud Computing is a recent paradigm for distributed computing as a service where dynamically scalable and often virtualized resources are provided over the Internet; it typically incorporates combinations of infrastructure as a service, platform as a service, and software as a service [209]. As the paradigm is relatively new, so are the considerations of its connections to semantic technologies: Reference [210] suggests an ontological clarification of basic cloud computing concepts. As the paradigm poses high requirements to cloud service providers, internal knowledge management as well as automated, rule-based system management may be useful. First ideas in this direction have already been communicated by leading-edge companies (like <http://www.fluidops.com/>). It should also be noted that besides the potential benefits of semantic technologies for cloud computing, it is also a thrilling question, which potentials cloud computing could offer to the Semantic Web [211].

Regarding the state of practice in Semantic Middleware, Semantic Web Services (SWS) are emphasized here, and the reader is pointed to a set of literature about SWS application prototypes [17, 186, 212]. Nevertheless, SWS are not yet as far in the practical, commercial take-up as, for instance, semantic search (see also [213, 214]) is. More convincing application domains seem to be those where serious information-integration challenges have already been a problem for a long time, where standardization and metadata-based approaches are already well-known, and where Web Services are already a widespread technology – examples comprise geospatial information provision [215, 216], health-care services [217, 218], and eGovernment [219, 220].

13.3.2.6 Semantic Software Engineering

Closely related to the idea of semantic middleware is the idea of Semantic Software Engineering, which is thus far not so deeply investigated or well-developed. By definition, the unique tasks of Software Engineering are more about software-development time than about runtime. But also there, the same questions arise, for example, *searching* for software components to be reused, thinking about what changes to make for *integrating* with other components, etc. However, in contrast to completely open and dynamic Web scenarios, several challenges for Semantic Web approaches (like standards enforcement, incentives, trust, security, provenance) may be easier to address in the “controlled environment” of organization-internal software development. The potential role of ontologies and semantic technologies in SE is surveyed in [221, 222]. The classification from [223] is depicted in  Fig. 13.9:



■ Fig. 13.9

Ontologies in Software Engineering (according to Happel and Seedorf)

Ontology-driven development (ODD) is about using ontologies at development time for describing the problem domain of the software to be developed. Prime examples are the approaches in the context of model-driven development (MDD) [224]. Benefits from using ontologies comprise reduced language ambiguity and automated validation and consistency checking.

Ontology-enabled development (OED) uses ontologies at development time for supporting software engineers with their tasks. For example, for the *identification of software components as reuse candidates* (component search), practically all aspects of semantic search can come into play (see [225] for an example). Reference [226] describes the Open Source software-development process with three ontologies, about (1) code, (2) bugs, and (3) interactions between members of the developer community. Such interactions are centered around artifacts, which may automatically be cross-linked through metadata; this supports better finding and the proactive recommendation of interesting information, for example, for bug resolution. In the same spirit, recent developments comprise, for example, the **baetle** ontology for software bugs and bug tracking systems as well as several endeavors in semantic search for bug tracking information on the Web [134], or the **EvoOnt** software-evolution OWL ontology which – together with the **iSPARQL** similarity-aware ontology query language – can solve several software analysis tasks, such as the assessment of the amount of change between releases, the computation of software design metrics, or the detection of “code smells.” Reference [227] aggregates manifold feeds published by the different tools of a software forge. To this end, collected data are semantically reformatted into

RDF, Dublin Core, DOAP, and FOAF. The resulting semantic data can then be processed, republished, or displayed to project members in order to visualize and analyze a live picture of community activities in Open Source software development.

Ontology-based architectures (OBA) use an ontology as a primary artifact at runtime.

The ontology makes up a central part of the application logic. Business rule approaches are an example for this kind of application, or rule-based systems in general.

Ontology-enabled architectures (OEA) leverage ontologies to provide infrastructure support at the runtime of a software system. This is the case for all semantic middleware approaches listed above.

13.3.3 Selected Application Domains

After this functionality-centered description, some examples are provided for application domains with many ontology-based Semantic Web applications.

13.3.3.1 Ontologies About Cultural Heritage

The cultural heritage area is a domain with a huge amount of publicly available data and information, interested in being found, such that there are already long-standing metadata standardization efforts (see also [▶ Semantic Web Services](#)). It provides representational challenges, as almost all statements have some time and spatial aspect, which is not easy to represent and reason about. In [150], an overview of *semantic portals for cultural heritage* is given, including important vocabularies and ontologies in the area, examples for logical rules in cultural heritage KBs, and typical services of cultural heritage portals.

In this area, **CIDOC CRM** (Conceptual Reference Model) [228] is an ontology for terminology and information sharing in the cultural heritage area. As standard ISO 21127:2006, its scope is defined as the exchange and integration of heterogeneous scientific documentation relating to museum collections. The idea is that knowledge sharing between archives, libraries, and museums is facilitated. As a comprehensive metadata standard including the necessary top-level concepts for representing concrete historic events, places, people, etc., CIDOC CRM contains many concepts and complex relationships for, at the same time, representing aspects of a top-level ontology (e.g., regarding temporal reasoning), of an information ontology (describing complex metadata for artifacts), and of a domain ontology for the domain of historic and archeological and artistic artifacts (e.g., materials). There is an RDFS and an OWL version of CIDOC CRM.

In the **SCULPTEUR** project [229], existing database systems of several museums were mapped to CRM in order to allow cross-collection searching. A graph-based visualization of a simplified CRM model was the basis for browsing the metadata space with a concept browser tool. In order to provide a usable access to the huge space of instance data, the graph-based concept browser was accompanied by the mSpace multifaceted search approach (<http://www.mspace.fm/>). Besides the ontology-based search, SCULPTEUR

also offers a shape- and color-based similarity search on 2D images or 3D objects. The **REACH** project [230] also implements a hybrid ontology- and visual-based search in cultural heritage multimedia libraries. As above, annotation metadata are mapped to CIDOC CRM, and a hybrid search algorithm can combine evidence from similar visual characteristics and similar metadata. Reference [151] addresses two typical aspects of cultural heritage portals, namely, ontology-based spatiotemporal search as well as the visualization of concepts, but focus on the fact that many historical facts change over time (population of a town, borders of a country, names of a person, etc.) – they derive explicit time series of temporal part-of ontologies and visualize change over time. Interesting from the technological viewpoint is also [231]: The authors employ OWL ontologies and forward-chaining SWRL rules in a real-time ubiquitous application, which dynamically and in a context-aware manner delivers content to audio museum guides for people who walk through a museum exhibition. The selection also takes into account psychoacoustic properties of sound objects.

13.3.3.2 Ontologies in eGovernment

eGovernment is a demanding, as well as promising application area for semantic technologies [232, 233], be it for discovering, composing, and reconfiguring eGovernment Web Services [234, 235] and enabling semantic interoperability between public administrations' software [236], for knowledge management within public administrations [237], or for eGovernment information or service portals for citizen or companies [238].

OE-gov is an initiative of TopQuadrant Inc. to collect and distribute eGovernment OWL ontologies (<http://www.oegov.us/>). The importance of semantic interoperability in eGovernment has also been recognized by the European Union (<http://www.semic.eu/>), but did not yet lead to a widespread harmonization of ontologies and semantic resources. One of the earliest topics to be expected for standardization may be life-event ontologies, which are typically used for navigating through eGovernment service portals [239]. Recently, the Linked-Open-Data initiatives of the US and UK governments fueled significantly the interest in the LOD topic (see <http://www.data.gov/> and <http://data.gov.uk/>). More details on the relationship between the Semantic Web and eGovernment can be found in [▶ eGovernment](#).

13.3.3.3 Ontologies in the Life Science Domain

Probably the most important (and far developed) application domain of Semantic Web and ontology-based approaches is the area of life sciences, biotechnology, medicine, and pharmaceutical research and development [240].

Certainly the most notable ontology is the Gene Ontology (GO) [241] – worked on since the late 1990s – which currently comprises three structured controlled vocabularies (ontologies) that describe gene products in terms of their associated *biological processes*, *cellular components*, and *molecular functions* in a species-independent manner. The GO

project develops and maintains those ontologies, annotates gene products with respect to them, and also provides tools for maintenance and use. The Open Biomedical Ontologies (OBO) initiative [242] consortium is pursuing a strategy to overcome the problem of the proliferation of biomedical ontologies. Existing OBO ontologies, including the GO, are undergoing coordinated reform, and new ontologies are being created on the basis of an evolving set of shared principles governing ontology development. The result shall be an expanding family of ontologies designed to be interoperable and logically well-formed.

The ultimate purpose of all such efforts is demonstrated in the *BioPortal* [155], which gives the possibility for comfortably annotating all kinds of biomedical resources (such as gene expression datasets, descriptions of radiology images, clinical-trial reports, or PubMed article abstracts) with concepts from any publicly available biomedical ontology, and provides the user at query time with an integrated view on all these different resources. In 2008, BioPortal had already annotated more than 1.1 Mio elements (mostly PubMed article abstracts, but also, for example, more than 50,000 clinical-trial descriptions), on average each element annotated with 486 concepts.

Regarding *semantic integration* in life sciences, [243] go step-by-step through a real-world data integration example from biomedical research about Parkinson's Disease and illustrate with many examples how to design ontologies and how to build wrappers for a number of existing data sources from different biomedical research fields such that the whole information space can be queried jointly through SPARQL queries. References [154, 244] make a comprehensive analysis of *semantic search* in life sciences and present the **GoPubMed** and **GoWeb** system, respectively, which employ the Gene Ontology and the MeSH as background knowledge for improved question answering and document search. The **Conceptual Open Hypermedia Service (COHSE)** [245] provides *navigation between Web resources* in large, dynamic, and complex knowledge spaces, supported by an ontology as a conceptual model which, together with lexical labels, drives the dynamic linking of Web resources.

In the area of medical applications, one of the very first motivations for using ontologies is that of enabling message exchange between software, which uses different disease classification systems. Here, [246] discusses how the expressive power of OWL can be used to describe transformations between different encodings.

ASMER [247] is a deployed and operational system for active, semantic, *electronic medical records*. Electronic medical records allow one to have all patient data for one person integrated at one's fingertips. *Active semantic documents* are document collections automatically annotated with regard to one or more formal ontologies, including rules working on the semantic annotations and relationships for automatic and dynamic validation and decision-making support. In this respect, they combine aspects of *semantic information integration* and of *intelligent advisory systems*. In ASMER, medical information is annotated with regard to a comprehensive OWL ontology comprising aspects such as drugs and drug interactions, indications for drugs, medical conditions, treatments, diagnoses, and procedures. RDQL rules are then used, for example, for drug-interaction checks, for drug-formulary checks, for drug-dosage range checks, for drug-allergy interaction checks, etc. **HealthFinland** as an example for a *semantic portal* in the medical area has already been sketched above.

13.4 Related Resources

The following are key references for ontologies in Computer Science.

Handbook on Ontologies (2nd edition) [248] – The handbook on ontologies gives an extensive overview on ontologies as a topic in Computer Science. It covers foundations of ontology languages, engineering methods, and management infrastructure, and also addresses aspects of the application and usage of ontologies. Its content is strongly influenced by the field of the Semantic Web and, thus, provides beneficial further reading on the topics covered in this chapter. In its second edition, it has been updated with recent developments in technology and augmented by many evolving application areas of ontologies.

Foundations of Semantic Web Technologies [249] – A comprehensive textbook on the foundations of Semantic Web ontology languages. It provides a good basis for University courses on Semantic Web technology with a focus on the W3C-standardized ontology languages RDFS and OWL with their underlying formal logical semantics, including exercises and solutions. Besides the technical basics on ontology languages with their syntax and semantics, the book addresses aspects of reasoning with ontologies, ontology engineering, rule extensions and querying, as well as some applications.

The Description Logic Handbook (2nd Edition) [11] – A comprehensive overview on Description Logics (DLs) for symbolic knowledge representation. The book covers the foundations of various DLs with their formal, model-theoretic semantics, methods of automated reasoning in DLs, and various other topics.

Semantic Web for the Working Ontologist [250] – A recent introduction to modeling Semantic Web ontologies by means of the W3C standard languages RDFS and OWL. It conveys the principles of the Semantic Web, covers the essentials of the W3C Semantic Web ontology language stack, and addresses many methodological and practical aspects of ontology engineering. It is a good reference for an in-depth course in modeling with RDFS and OWL.

Ontology Learning and Population: Bridging the Gap between Text and Knowledge [31] – The book provides a survey on ontology learning techniques. It discusses ontologies for the Semantic Web, knowledge management, information retrieval, text clustering and classification, as well as natural language processing, all in the context of ontology learning.

Online resources: Besides literature, some computational resources should also be mentioned: There is a growing number of *ontology repositories* [251] with numerous reusable ontologies that is going to reduce the cold-start problem for ontology construction, for instance:

- **OntoSelect** with currently 1,530 ontologies in RDFS, DAML, and OWL [252]
- The US National Center for Biomedical Ontology's **BioPortal** [253] with 162 ontologies containing ca. 700,000 concepts
- The European Bioinformatics Institute's **Ontology Lookup Service OLS** hosting 69 ontologies with more than 880,000 terms
- The Australian **Pronto** repository with about 230 OWL, RDF, and OBO ontologies
- The **National Finnish Ontology Service ONKI**

- **SchemaWeb**, a directory of 240 RDF schemas expressed in the RDFS, OWL, and DAML+OIL schema languages

From the technological point of view, easier, reuse-based ontology engineering must be accompanied by a better understanding of *networked ontologies* as investigated in the NeOn project; NeOn also provided extensive sample ontologies and applications in the pharmaceutical and in the agricultural domains. Results from NeOn and other projects were made accessible through the Watson “Semantic Web Gateway” (<http://watson.kmi.open.ac.uk/>) that mainly provides software components and Application Programming Interfaces (APIs) for exploiting the Semantic Web knowledge. See [Semantic Web Search Engines](#) for more about Watson.

A recent idea for applying semantic technologies and foster their proliferation, as well, is the approach of *Linked Open Services* (<http://www.linkedopenservices.org/>): exposing services, that is functionalities, on the Web using the same technologies that are associated with linked data, in particular HTTP, RDF, and SPARQL. See [Semantic Web Services](#) for more on the relationship between linked data and Web Services.

Finally, one can get a good impression of up-to-date Semantic Web applications, from:

- The World Wide Web Consortium’s Best Practices and Deployment Working Group: <http://www.w3.org/2001/sw/BestPractices/>
- The Semantic Web Challenge: <http://challenge.semanticweb.org/> and
- The Semantic Web Service Challenge: <http://sws-challenge.org/>

13.5 Future Issues

This chapter presented ontologies as one of the major cornerstones of Semantic Web technology. The essential characteristics of ontologies in Computer Science were explained, based on a formal ontology model, which is able to express all typical ontological modeling constructs, while being independent from a concrete ontology language. Ontology-engineering methods and tools were surveyed, elaborating on tools and methods for manual ontology construction on the one hand, and tools and methods for ontology learning from text, on the other.

A number of basic benefits and generic functionalities of ontologies were listed, and typical ontology use cases were discussed, such as semantic search and semantic portals, semantic information integration and linked open data, intelligent advisory systems, as well as semantic middleware and software engineering. Many concrete examples for Semantic Web ontologies in important domains (like life sciences or cultural heritage) and their usage in one of these use-case categories were given. Many usage examples were given; nevertheless, there are still important application domains (e.g., eLearning, eScience [254], or geospatial and environmental information services) and use cases (e.g., *patent search* [255]) that were left out.

In general, there is a broad range of knowledge structures, which can be subsumed under the ontology label; consequently, also the applications differ, depending on the

question of which specific aspects of ontologies are exploited and whether the focus is more on heavyweight or on lightweight knowledge models. There is still scope for applied Computer Science to better understand which kind of ontology is a worthwhile investment in which kind of application. Already [256] pointed out the trade-offs between degree of formality, sharing scope, and stability of knowledge structures – which make certain kinds of ontologies more or less profitable. Reference [257] gives a more thorough analysis from an economic point of view and investigates not only technical, but also social, economic, and legal difficulties that constrain the space of practically possible ontologies. Some promising developments are seen toward overcoming such difficulties. A better understanding of where to use heavyweight and where to use lightweight ontologies (and how to use both together) is one such development; the advent of social efforts for ontology building and maintenance is another one – an interesting approach has been sketched above, the OBO initiative for the controlled evolution of ontologies in the community of bioinformatics researchers. The *knowledge maturing theory* of the **MATURE** project with its tools and methods for collaborative, usage-embedded ontology development is another one (e.g., the **SOBOLEO** tool [258]).

There is also a recognizable trend toward ontology reuse and leveraging structured resources in ontology development, which will grow stronger as more and more ontologies and RDF repositories become available. This way of bootstrapping Semantic Web content creation could significantly speed up the formalization of metadata on the Web and foster semantic interoperability between vocabularies.

The growing amount of semantic metadata in the Web as well as the growing “Web of Data” boosted by the LOD initiative fueled a renewed and increased interest in Semantic Search and Semantic Web Search Engines. Taking into account the scalability requirements of web-scale solutions as well as other, related particularities (e.g., imprecise queries or inconsistent data sources), systems like **Semplore** [259] reconsider many of the issues already discussed in [▶ Sect. 3.2.1](#). Regarding “query disambiguation,” for instance, [260] investigates how informal, keyword-like user queries can be transformed into query-graphs that easily lead to SPARQL queries. Later processing steps comprise aspects like “query decomposition and planning,” which were presented above in [▶ Sect. 3.2.2](#). Regarding post-processing of answers, semantics-aware ranking of results are being developed. The sheer mass of data also makes necessary completely new work directions (of course, not so new in the areas like databases or Internet search engines) such as continuous, efficient index update in a highly dynamic world, or top-k retrieval for delivering ranked results.

Regarding longer-term prospects, it is expected that the coming years will see a new wealth of applications coming from the fact that real world and virtual worlds are increasingly interwoven. In the area of ambient intelligence, assisted living, etc., *context-aware, pervasive computing* will open up new opportunities [261] for intelligent software, based on semantically rich notions of context; combining this with the approach of Semantic Web Services, *service-oriented context-aware middleware* may become a next software paradigm [262] – where semantic technologies are needed to deal with the complexity of such hardware–software landscapes, and to inject common-sense

intelligence into the processing algorithms. For instance, the **openAAL** initiative suggests an Open Source context-aware middleware for Ambient Assisted Living: <http://openaal.org/>. The recent notion of a *Semantic Sensor Web* [263, 264] goes in the same direction and further emphasizes the need for common-sense knowledge on one hand, a great variety of potential applications on the other, and also completely new requirements with regard to scalability and efficiency of reasoning.

13.6 Cross-References

- Knowledge Management in Large Organizations
- KR and Reasoning on the Semantic Web: OWL
- Semantic Web Architecture

Acknowledgments

This work has been partially financed by the German Federal State of Baden-Württemberg, by the Federal German Ministry of Economics and Technology under the research program THESEUS (“New Technologies for the Internet of Services”), by the Deutsche Forschungsgemeinschaft (DFG) under the project Multipla (“Multi Ontology Learning Crossing the Boundaries of Domains and Languages”), and by the European Commission through projects NeOn (“Lifecycle Support for Networked Ontologies”) and SEALS (“Semantic Evaluation at Large Scale”). We also acknowledge financial support for Johanna Völker through a Margarete-von-Wrangell scholarship awarded by the European Social Fund (ESF) and the Ministry of Science, Research, and the Arts Baden-Württemberg.

References

1. Guarino, N., Giaretta, P.: Ontologies and knowledge bases: towards a terminological clarification. In: Mars, N.J.I. (ed.) *Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing*, pp. 25–32. IOS Press, Amsterdam (1995)
2. Craig, E.: *Ontology*. In: Craig, E. (ed.) *Routledge Encyclopedia of Philosophy*, pp. 117–118. Routledge, New York (1998)
3. Studer, R., Benjamins, V.R., Fensel, D.: *Knowledge engineering: principles and methods*. *IEEE Trans. Knowl. Data Eng.* **25**(1–2), 161–197 (1998)
4. Guarino, N.: *Formal ontology and information systems*, preface. In: Guarino, N. (ed.) *Proceedings of the First International Conference on Formal Ontologies in Information Systems (FOIS 1998)*, Trento, pp. 3–15. IOS Press, Amsterdam (1998)
5. Grimm, S.: *A unifying formal ontology model*. In: *Proceedings of the International Conference on Knowledge Engineering and Ontology Design (KEOD 2009)*, Funchal (2009)
6. Sowa, J.F.: *Knowledge Representation*. Brooks Cole, Pacific Grove (2000)
7. Russel, S., Norvig, P.: *Artificial Intelligence – A Modern Approach*. Prentice-Hall, New Jersey (1995)
8. Klyne, G., Carroll, J.: *RDF concepts and abstract syntax*. <http://www.w3.org/TR/rdfprimer/> (2004). Accessed 4 Aug 2010
9. Brickley, D., Guha, R.V.: *RDF vocabulary description language – RDF schema*. <http://www.w3.org/TR/rdf-schema/> (2004)

10. Horrocks, I., Patel-Schneider, P.F., van Harmelen, F.: From SHIQ and RDF to OWL: the making of a web ontology language. *J. Web Semant.* **1**(1), 7–26 (2003)
11. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): *The Description Logic Handbook*. Cambridge University Press, Cambridge (2003)
12. Kifer, M., Lausen, G., Wu, J.: Logical foundations of object-oriented and frame-based languages. *J. ACM* **42**(4), 741–843 (1995)
13. Ullman, J.D.: *Principles of Database and Knowledge-Base Systems*, vol. I/II. Computer Science Press, New York (1989)
14. Motik, B.: Reasoning in description logics using resolution and deductive databases. Ph.D. thesis, Universität Karlsruhe (TH) (2006)
15. Krötzsch, M., Rudolph, S., Hitzler, P.: Description logic rules. In: *Proceedings of the 18th European Conference on Artificial Intelligence (ECAI 2008)*, Patras, pp. 80–84. IOS Press, Amsterdam (2008)
16. de Bruijn, J., Lausen, H., Polleres, A., Fensel, D.: The web service modeling language WSM: an overview. In: *Proceedings of the Third European Semantic Web Conference (ESWC 2006)*, Budva. *Lecture Notes in Computer Science*, vol. 4011, pp. 590–604. Springer, Berlin (2006)
17. Studer, R., Grimm, S., Abecker, A. (eds.): *Semantic Web Services: Concepts, Technologies, and Applications*. Springer, Berlin (2007)
18. Miles, A., Bechhofer, S.: SKOS simple knowledge organization system reference. <http://www.w3.org/TR/skos-reference> (2008). Accessed 4 Aug 2010
19. Fellbaum, C.D.: *WordNet – An Electronic Lexical Database*. Language, Speech, & Communication. MIT Press, Cambridge (1998)
20. Golder, S., Huberman, B.A.: Usage patterns of collaborative tagging systems. *J. Inf. Sci.* **32**(2), 198–208 (2006)
21. Oltramari, A., Gangemi, A., Guarino, N., Masolo, C.: Sweetening ontologies with DOLCE. In: Gómez-Pérez, A., Benjamins, V.R. (eds.) *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2002)*, Siguenza. *Lecture Notes in Computer Science*, vol. 2473, pp. 223–233. Springer, Heidelberg (2002)
22. Niles, I., Pease, A.: Towards a standard upper ontology. In: Welty, C., Smith, B. (eds.) *Proceedings of the Second International Conference on Formal Ontology in Information Systems (FOIS 2001)*, Ogunquit (2001)
23. Hepp, M.: Products and services ontologies: a methodology for deriving OWL ontologies from Industrial Categorization Standards. *Int. J. Semant. Web Inf. Syst.* **2**(1), 72–99 (2006)
24. Hepp, M.: GoodRelations: an ontology for describing products and services offers on the web. In: Gangemi, A., Euzenat, J. (eds.) *Proceedings of the 16th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2008)*, Acitrezza. *Lecture Notes in Computer Science*, vol. 5268, pp. 329–346. Springer, Heidelberg (2008)
25. Roman, D., Keller, U., Lausen, H., Lara, R., de Bruijn, J., Stollberg, M., Polleres, A., Feier, C., Bussler, C., Fensel, D.: Web service modeling ontology. *J. Appl. Ontol.* **1**(1), 77–106 (2005)
26. Bojars, U., Heitmann, B., Oren, E.: A prototype to explore content and context on social community sites. In: *SABRE Conference on Social Semantic Web (CSSW 2007)*, Leipzig (2007)
27. Guo, Y., Pan, Z., Heflin, J.: An evaluation of knowledge base systems for large OWL datasets. In: Plexousakis, D., McIlraith, S., van Harmelen, F. (eds.) *Proceedings of the Third International Semantic Web Conference (ISWC 2004)*, Hiroshima. *Lecture Notes in Computer Science*, vol. 3298, pp. 274–288. Springer, Heidelberg (2004)
28. Ma, L., Yang, Y., Qiu, Z., Tong Xie, G., Pan, Y., Liu, S.: Towards a complete OWL ontology benchmark. In: *Proceedings of the Third European Semantic Web Conference (ESWC 2006)*, Budva. *Lecture Notes in Computer Science*, vol. 4011, pp. 125–139. Springer, Heidelberg (2006)
29. Mizoguchi, R., Kozaki, K.: Ontology engineering environments. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*. International Handbooks on Information Systems, 2nd edn, pp. 315–336. Springer, Heidelberg (2009)
30. Sure, Y., Staab, S., Studer, R.: Ontology engineering methodology. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*. International Handbooks on Information Systems, 2nd edn, pp. 135–152. Springer, Heidelberg (2009)

31. Buitelaar, P., Cimiano, P. (eds.): *Ontology Learning and Population: Bridging the Gap between Text and Knowledge*. *Frontiers in Artificial Intelligence and Applications*, vol. 167. IOS Press, Amsterdam (2008)
32. Jaimes, A., Smith, J.R.: Semi-automatic, data-driven construction of multimedia ontologies. In: *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME 2003)*, Baltimore, pp. 781–784. IEEE Computer Society, Washington, DC (2003)
33. Newbold, N., Vrusias, B., Gillam, L.: Lexical ontology extraction using terminology analysis: automating video annotation. In: *European Language Resources Association (ELRA) (ed.) Proceedings of the Sixth International Language Resources and Evaluation (LREC 2008)*, Marrakech (2008)
34. Jäschke, R., Hotho, A., Schmitz, C., Ganter, B., Stumme, G.: Discovering shared conceptualizations in Folksonomies. *J. Web Semant.* **6**(1), 38–53 (2008)
35. Gómez-Pérez, A., Corcho-García, O., Fernández-López, M.: *Ontological Engineering*. Springer, New York (2003)
36. Rector, A., Drummond, N., Horridge, M., Rogers, J., Knublauch, H., Stevens, R., Wang, H., Wroe, C.: OWL pizzas: practical experience of teaching OWL DL – common errors & common patterns. In: *Motta, E., Shadbolt, N., Stutt, A., Gibbins, N. (eds.) Proceedings of the 14th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2004)*, Whittlebury Hall. *Lecture Notes in Computer Science*, vol. 3257, pp. 63–81. Springer, Heidelberg (2004)
37. Day-Richter, J., Harris, M.A., Haendel, M., Lewis, S.: OBO-edit – an ontology editor for biologists. *Bioinformatics* **23**(16), 2198–2200 (2007)
38. Brockmans, S., Haase, P., Hitzler, P., Studer, R.: A metamodel and UML profile for rule-extended OWL DL ontologies. In: *Sure, Y., Domingue, J. (eds.) The Semantic Web: Research and Applications*. *Lecture Notes in Computer Science*, vol. 4011, pp. 303–316. Springer, Berlin (2006)
39. Bernstein, A., Kaufmann, E.: GINO – a guided input natural language ontology editor. In: *Cruz, I.F., et al. (eds.) Proceedings of the Fifth International Semantic Web Conference (ISWC 2006)*, Athens, GA. *Lecture Notes in Computer Science*, vol. 4273, pp. 144–157. Springer, Heidelberg (2006)
40. Davis, B., Iqbal, A.A., Funk, A., Tablan, V., Bontcheva, K., Cunningham, H., Handschuh, S.: RoundTrip ontology authoring. In: *Proceedings of the Seventh International Semantic Web Conference (ISWC 2008)*, Karlsruhe. *Lecture Notes in Computer Science*, vol. 5318, pp. 50–65. Springer, Berlin (2008)
41. Fuchs, N.E., Kaljurand, K., Schneider, G.: Attempto controlled English meets the challenges of knowledge representation, reasoning, interoperability and user interfaces. In: *Sutcliffe, G.C.J., Goebel, R.G. (eds.) Proceedings of the 19th Florida Artificial Intelligence Research Society Conference (FLAIRS 2006)*, Melbourne Beach, pp. 664–669. AAAI Press, Menlo Park (2006)
42. Hart, G., Johnson, M., Dolbear, C.: Rabbit: developing a control natural language for authoring ontologies. In: *Proceedings of the Fifth European Semantic Web Conference (ESWC 2008)*, Tenerife. *Lecture Notes in Computer Science*, vol. 5021, pp. 348–360. Springer, Berlin (2008)
43. Krötzsch, M., Vrandečić, D., Völkel, M., Haller, H., Studer, R.: Semantic wikipedia. *J. Web Semant.* **5**, 251–261 (2007)
44. Tudorache, T., Noy, N., Tu, S., Musen, M.A.: Supporting collaborative ontology development in protégé. In: *Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T.W., Thirunarayan, K. (eds.) Proceedings of the Seventh International Semantic Web Conference (ISWC 2008)*, Karlsruhe. *Lecture Notes in Computer Science*, vol. 5318, pp. 17–32. Springer, Berlin (2008)
45. Wennerberg, P., Zillner, S., Möller, M., Buitelaar, P., Sintek, M.: KEMM: a knowledge engineering methodology in the medical domain. In: *Eschenbach, C., Grüninger, M. (eds.) Proceedings of the Fifth International Conference on Formal Ontology in Information Systems (FOIS 2008)*, Saarbrücken. *Frontiers in Artificial Intelligence and Applications*, Saarbrücken, vol. 183, pp. 79–91. IOS Press, Amsterdam (2008)
46. Guarino, N., Welty, C.: An overview of OntoClean. In: *Staab, S., Studer, R. (eds.) Handbook on Ontologies*. *International Handbooks*

- on Information Systems, 2nd edn, pp. 245–267. Springer, Heidelberg (2009)
47. Simperl, E., Tempich, C.: Exploring the economical aspects of ontological engineering. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*. International Handbooks on Information Systems, 2nd edn, pp. 337–358. Springer, Heidelberg (2009)
 48. Tempich, C., Simperl, E., Luczak, M., Studer, R., Pinto, H.S.: Argumentation-based ontology engineering. *IEEE Intell. Syst.* **22**(6), 52–59 (2007)
 49. Vrandečić, D.: Ontology evaluation. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*. International Handbooks on Information Systems, 2nd edn, pp. 293–313. Springer, Heidelberg (2009)
 50. Gangemi, A., Presutti, V.: Ontology design patterns. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*. International Handbooks on Information Systems, 2nd edn, pp. 221–243. Springer, Heidelberg (2009)
 51. Presutti, V., Gangemi, A.: Content ontology design patterns as practical building blocks for web ontologies. In: *Proceedings of the 27th International Conference on Conceptual Modeling (ER 2008)*, Barcelona, pp. 128–141. Springer, Berlin (2008)
 52. Scharffe, F., Fensel, D.: Correspondence patterns for ontology alignment. In: Gangemi, A., Euzenat, J. (eds.) *Proceedings of the 16th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2008)*, Acitrezza. *Lecture Notes in Computer Science*, vol. 5268, pp. 83–92. Springer, Heidelberg (2008)
 53. Fernández-López, M., Gómez-Pérez, A., Sierra, J.P., Sierra, A.P.: Building a chemical ontology using methontology and the ontology design environment. *IEEE Intell. Syst.* **14**(1), 37–46 (1999)
 54. Tempich, C., Pinto, H.S., Sure, Y., Staab, S.: An argumentation ontology for DIstributed, Loosely-controlled and evolvInG Engineering processes of oNTologies (DILIGENT). In: Gómez-Pérez, A., Euzenat, J. (eds.) *Proceedings of the Second European Semantic Web Conference (ESWC 2005)*, Heraklion. *Lecture Notes in Computer Science*, vol. 3532, pp. 241–256. Springer, Berlin (2005)
 55. Noy, N., McGuinness, D.L.: *Ontology development 101: a guide to creating your first ontology*. Technical report KSL-01-05 and SMI-2001-0880, Stanford Knowledge Systems Laboratory and Stanford Medical Informatics (2001)
 56. Sure, Y., Studer, R.: On-to-knowledge methodology. In: Davies, J., Fensel, D., van Harmelen, F. (eds.) *On-To-Knowledge: Semantic Web enabled Knowledge Management*, Chapter. 3, pp. 33–46. Wiley, New York (2002)
 57. De Nicola, A., Missikoff, M., Navigli, R.: A software engineering approach to ontology building. *Inf. Syst.* **34**(2), 258–275 (2009)
 58. Kotis, K., Vouros, A.: Human-centered ontology engineering: the HCOME methodology. *Knowl. Inf. Syst.* **10**(1), 109–131 (2006)
 59. Suárez-Figueroa, M.C., et al.: D5.4.1 NeOn methodology for building contextualized ontology networks. Technical report 5.4.1, NeOn deliverable (2009)
 60. Jarrar, M., Meersman, R.: Ontology engineering – the DOGMA approach. In: Dillon, T.S., Chang, E., Meersman, R., Sycara, K. (eds.) *Advances in Web Semantics I*. *Lecture Notes in Computer Science*, vol. 4891. Springer, Berlin (2008)
 61. Grüniger, M., Fox, M.: Methodology for the design and evaluation of ontologies. In: *Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing at International Joint Conference on Artificial Intelligence (IJCAI 1995)*, Montreal (1995)
 62. Uschold, M., King, M.: Towards a methodology for building ontologies. In: *Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing at International Joint Conference on Artificial Intelligence (IJCAI 1995)*, Montreal (1995)
 63. Garcia Castro, A., Rocca-Serra, P., Stevens, R., Taylor, C., Nashar, K., Ragan, M.A., Sansone, S.-A.: The use of concept maps during knowledge elicitation in ontology development processes—the nutrigenomics use case. *BMC Bioinform.* **7**, 267 (2006)
 64. Noy, N., Fergerson, R., Musen, M.: The knowledge model of protégé-2000: combining interoperability and flexibility. In: Dieng, R., Corby, O. (eds.) *Proceedings of the 12th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2000)*, Juom-les-pins. *Lecture Notes in Artificial Intelligence*, vol. 1937, pp. 17–32. Springer, Berlin (2000)

65. Arpírez, J.C., Corcho, O., Fernández-López, M., Gómez-Pérez, A.: WebODE: a scalable workbench for ontological engineering. In: Proceedings of the First International Conference on Knowledge Capture (K-CAP 2001), Victoria, pp. 6–13. ACM, New York (2001)
66. Sure, Y., Angele, J., Staab, S.: OntoEdit: multifaceted inferencing for ontology engineering. *J. Data Semant. Lecture Notes in Computer Science*, **2800**, 128–152 (2003)
67. Völker, J., Vrandečić, D., Sure, Y., Hotho, A.: AEON – an approach to the automatic evaluation of ontologies. *J. Appl. Ontol.* **3** (1–2), 41–62, ISSN 1570-5838. IOS Press, Amsterdam (2008)
68. Cimiano, P., Mädche, A., Staab, S., Völker, J.: Ontology learning. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*. International Handbooks on Information Systems, 2nd edn, pp. 245–267. Springer, Heidelberg (2009)
69. Grimnes, G.A., Edwards, P., Preece, A.D.: Learning meta-descriptions of the FOAF network. In: McIlraith, S.A., et al. (eds.) *Proceedings of the Third International Semantic Web Conference (ISWC 2004)*, Hiroshima. *Lecture Notes in Computer Science*, vol. 3298, pp. 152–165. Springer, Berlin (2004)
70. Basili, R., Moschitti, A., Paziienza, M.T., Zanzotto, F.M.: A contrastive approach to term extraction. In: *Proceedings of the Fourth Terminology and Artificial Intelligence Conference (TIA 2001)*, Nancy, pp. 119–128 (2001)
71. Drouin, P.: Detection of domain specific terminology using corpora comparison. In: *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004)*, pp. 79–82. European Language Resources Association, Lisbon (2004)
72. Kilgarriff, A.: Comparing corpora. *Int. J. Corpus Linguist.* **1**(6), 97–103 (2001)
73. Brunzel, M.: The XTREEM methods for ontology learning from web documents. In: Buitelaar, P., Cimiano, P. (eds.) *Ontology Learning and Population: Bridging the Gap between Text and Knowledge*. *Frontiers in Artificial Intelligence and Applications*, vol. 167, pp. 3–26. IOS Press, Amsterdam (2008)
74. Jacquemin, C.: *Spotting and Discovering Terms Through Natural Language Processing*. MIT Press, Cambridge (2001)
75. Witschel, H.F.: Terminology extraction and automatic indexing – comparison and qualitative evaluation of methods. In: Madsen, B.N., Thomsen, H.E. (eds.) *Proceedings of the Seventh International Conference on Terminology and Knowledge Engineering (TKE 2005)*. Association for Terminology and Knowledge Transfer, Copenhagen (2005)
76. Harris, Z.S.: Distributional structure. *Word* **10**(23), 146–162 (1954)
77. Pantel, P., Crestan, E., Borkovsky, A., Popescu, A.-M., Vyas, V.: Web-scale distributional similarity and entity set expansion. In: *Proceedings of the International Conference on Empirical Methods in Natural Language Processing (EMNLP 2009)*, Singapore, pp. 938–947 (2009)
78. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: *Proceedings of the 14th International Conference on Computational Linguistics (COLING 1992)*, Nantes, pp. 539–545 (1992)
79. Ogata, N., Collier, N.: Ontology express: statistical and non-monotonic learning of domain ontologies from text. In: *Proceedings of the Workshop on Ontology Learning and Population (OLP 2004) at the European Conference on Artificial Intelligence (ECAI 2004)*, Valencia (2004)
80. Cimiano, P., Handschuh, S., Staab, S.: Towards the self-annotating web. In: *Proceedings of the 13th International World Wide Web Conference (WWW 2004)*, Manhattan, pp. 462–471. ACM, New York (2004)
81. Potrich, A., Pianta, E.: L-ISA: learning domain specific isa-relations from the web. In: *European Language Resources Association (ed.) Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech (2008)
82. Blohm, S., Cimiano, P.: Using the web to reduce data sparseness in pattern-based information extraction. In: *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2007)*, Bled, pp. 18–29. Springer, Heidelberg (2007)
83. Cimiano, P., Ladwig, G., Staab, S.: Gimme the context: context-driven automatic semantic annotation with C-PANKOW. In: Ellis, A., Hagino, T. (eds.) *Proceedings of the 14th International World Wide Web Conference*

- (WWW 2005), Chiba, pp. 332–341. ACM, New York (2005)
84. Keller, F., Lapata, M., Ourioupina, O.: Using the web to overcome data sparseness. In: Hajič, J., Matsumoto, Y. (eds.) *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, Philadelphia, pp. 230–237. Association for Computational Linguistics (2002)
 85. Resnik, P., Smith, N.A.: The web as a parallel corpus. *Comput. Linguist.* **29**(3), 349–380 (2003)
 86. Blohm, S., Cimiano, P., Stemle, E.: Harvesting relations from the web – quantifying the impact of filtering functions. In: *Proceedings of the 22nd Conference on Artificial Intelligence (AAAI 2007)*, Vancouver, pp. 1316–1323. AAAI Press (2007)
 87. Brewster, C., Iria, J., Zhang, Z., Ciravegna, F., Guthrie, L., Wilks, Y.: Dynamic iterative ontology learning. In: *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2007)*, Borovets (2007)
 88. Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Web-scale information extraction in KnowItAll (preliminary results). In: *Proceedings of the 13th International World Wide Web Conference (WWW 2004)*, Manhattan, pp. 100–109. ACM, New York (2004)
 89. Pantel, P., Pennacchiotti, M.: Espresso: leveraging generic patterns for automatically harvesting semantic relations. In: *Proceedings of the 21st International Conference on Computational Linguistics (COLING 2006) and 44th Annual Meeting of the Association for Computational Linguistics (ACL 2006)*, Sydney, pp. 113–120. The Association for Computer Linguistics (2006)
 90. Caraballo, S.A.: Automatic construction of a hypernym-labeled noun hierarchy from text. In: *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL 1999)*, Maryland, pp. 120–126. Association for Computational Linguistics (1999)
 91. Cimiano, P., Hotho, A., Staab, S.: Comparing conceptual, divide and agglomerative clustering for learning taxonomies from text. In: López de Mántaras, R., Saitta, L. (eds.) *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI 2004)*, Valencia, pp. 435–439. IOS Press, Amsterdam (2004)
 92. Faure, D., Nédellec, C.: Knowledge acquisition of predicate argument structures from technical texts using machine learning: the system ASIUM. In: Fensel, D., Studer, R. (eds.) *Proceedings of the 11th International Conference on Knowledge Engineering and Knowledge Management (EKAW 1999)*, Dagstuhl Castle. Lecture Notes in Computer Science, vol. 1621, pp. 329–334. Springer, Berlin (1999)
 93. Cimiano, P., Völker, J.: Towards large-scale, open-domain and ontology-based named entity classification. In: Angelova, G., Bontcheva, K., Mitkov, R., Nicolov, N. (eds.) *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2005)*, Borovets, pp. 166–172. INCOMA, Bulgaria (2005)
 94. Coppola, B., Gangemi, A., Gliozzo, A.M., Picca, D., Presutti, V.: Frame detection over the semantic web. In: Aroyo, L., Traverso, P., Ciravegna, F., Cimiano, P., Heath, T., Hyvönen, E., Mizoguchi, R., Oren, E., Sabou, M., Simperl, E.P.B. (eds.) *Proceedings of the Sixth European Semantic Web Conference (ESWC 2009)*. Lecture Notes in Computer Science, vol. 5554, pp. 126–142. Springer, Berlin (2009)
 95. Suchanek, F.M., Sozio, M., Weikum, G.: SOFIE: a self-organizing framework for information extraction. In: *Proceedings of the 18th International Conference on World Wide Web (WWW 2009)*, Madrid, pp. 631–640. ACM, New York (2009)
 96. Etzioni, O., Banko, M., Soderland, S., Weld, D.S.: Open information extraction from the web. *Commun. ACM* **51**(12), 68–74 (2008)
 97. Schutz, A., Buitelaar, P.: RelExt: a tool for relation extraction from text in ontology extension. In: Gil, Y., Motta, E., Benjamins, V.R. (eds.) *Proceedings of the Fourth International Semantic Web Conference (ISWC 2005)*, Galway. Lecture Notes in Computer Science, vol. 3729, pp. 593–606. Springer, Berlin (2005)
 98. Cimiano, P., Hartung, M., Ratsch, E.: Finding the appropriate generalization level for binary relations extracted from the Genia Corpus. In: *Proceedings of the Fifth International Conference on Language Resources and Evaluation*

- (LREC 2006), Genoa, pp. 161–169. European Language Resources Association (2006)
99. Heyer, G., Läuter, M., Quasthoff, U., Wittig, T., Wolff, C.: Learning relations using collocations. In: Mädche, A., Staab, S., Nédellec, C., Hovy, E.H. (eds.) Proceedings of the Second Workshop on Ontology Learning (OL 2001) at International Joint Conference on Artificial Intelligence (IJCAI 2001), Seattle, vol. 38. CEUR-WS.org (2001)
 100. Antunes, C.: Mining rules in the Onto4AR framework for updating domain ontologies. In: Proceedings of the IADIS European Conference on Data Mining (ECDM 2007), Lisbon, pp. 95–101. IADIS Press (2007)
 101. Mädche, A., Staab, S.: Discovering conceptual relations from text. In: Horn, W. (ed.) Proceedings of the 14th European Conference on Artificial Intelligence (ECAI 2000), Berlin, pp. 321–325. IOS Press, Amsterdam (2000)
 102. Kavalec, M., Svátek, V.: A study on automated relation labelling in ontology learning. In: Buitelaar, P., Cimiano, P., Magnini, B. (eds.) *Ontology Learning from Text: Methods, Evaluation and Applications*. Frontiers in Artificial Intelligence and Applications, vol. 123, pp. 44–58. IOS Press, Amsterdam (2005)
 103. Rudolph, S.: Acquiring generalized domain-range restrictions. In: Medina, R., Obiedkov, S. (eds.) Proceedings of the Sixth International Conference on Formal Concept Analysis (ICFCA 2008), Montreal. Lecture Notes in Artificial Intelligence, vol. 4933, pp. 32–45. Springer, Heidelberg (2008)
 104. Völker, J., Rudolph, S.: Fostering web intelligence by semi-automatic OWL ontology refinement. In: Proceedings of the Seventh International Conference on Web Intelligence (WI 2008), Sydney. IEEE Washington, DC (2008)
 105. Völker, J., Vrandečić, D., Sure, Y., Hotho, A.: Learning disjointness. In: Franconi, E., Kifer, M., May, W. (eds.) Proceedings of the Fourth European Semantic Web Conference (ESWC 2007), Innsbruck. Lecture Notes in Computer Science, vol. 4519, pp. 175–189. Springer, Berlin (2007)
 106. Fanizzi, N., Iannone, L., Palmisano, I., Semeraro, G.: Concept formation in expressive description logics. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) Proceedings of the 15th European Conference on Machine Learning (ECML). Lecture Notes in Artificial Intelligence, vol. 3201. Springer, Berlin (2004)
 107. Hellmann, S., Lehmann, J., Auer, S.: Learning of OWL class descriptions on very large knowledge bases. *Int. J. Semant. Web Inf. Syst.* 5(2), 25–48 (2009)
 108. Baader, F., Ganter, B., Sertkaya, B., Sattler, U.: Completing description logic knowledge bases using formal concept analysis. In: Veloso, M.M. (ed.) Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), Hyderabad, pp. 230–235. AAAI Press, Menlo Park (2007)
 109. Völker, J., Rudolph, S.: Lexico-logical acquisition of OWL DL axioms – an integrated approach to ontology refinement. In: Medina, R., Obiedkov, S. (eds.) Proceedings of the Sixth International Conference on Formal Concept Analysis (ICFCA 2008), Montreal. Lecture Notes in Artificial Intelligence, vol. 4933, pp. 62–77. Springer, Berlin (2008)
 110. Shamsfard, M., Barforoush, A.A.: Learning ontologies from natural language texts. *Int. J. Hum. Comput. Stud.* 60(1), 17–63 (2004)
 111. Völker, J.: Learning expressive ontologies. *Studies on the semantic web*, vol. 2. AKA Verlag/IOS Press, Heidelberg (2009)
 112. d’Aquin, M., Motta, E., Dzbor, M., Gridinoc, L., Heath, T., Sabou, M.: Collaborative semantic authoring. *IEEE Intell. Syst.* 23(3), 80–83 (2008)
 113. Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T.M., Nigam, K., Slattery, S.: Learning to construct knowledge bases from the world wide web. *Artif. Intell.* 118(1–2), 69–113 (2000)
 114. Mädche, A., Volz, R.: The text-to-onto ontology extraction and maintenance system. In: Proceedings of the Workshop on Integrating Data Mining and Knowledge Management at the First International Conference on Data Mining (ICDM 2001), San Jose (2001)
 115. Buitelaar, P., Olejnik, D., Sintek, M.: A protégé plug-in for ontology extraction from text based on linguistic analysis. In: Proceedings of the First European Semantic Web Symposium (ESWS 2004), Heraklion. Lecture Notes in Computer Science, vol. 3053, pp. 31–44. Springer, Berlin (2004)

116. Morita, T., Shigeta, Y., Sugiura, N., Fukuta, N., Izumi, N., Yamaguchi, T.: DOODLE-OWL: OWL-based semi-automatic ontology development environment. In: *Proceedings of the Workshop on Evaluation of Ontology-Based Tools (EON 2004) at the International Semantic Web Conference (ISWC 2004)*, Hiroshima (2004)
117. Cimiano, P., Völker, J.: Text2Onto – a framework for ontology learning and data-driven change discovery. In: Montoyo, A., Munoz, R., Metais, E. (eds.) *Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB 2005)*, Alicante. *Lecture Notes in Computer Science*, vol. 3513, pp. 227–238. Springer, Berlin (2005)
118. Velardi, P., Navigl, R., Cucchiarelli, A., Neri, F.: Evaluation of OntoLearn, a methodology for automatic learning of domain ontologies. In: Buitelaar, P., Cimiano, P., Magnini, B. (eds.) *Ontology Learning from Text: Methods, Evaluation and Applications*. *Frontiers in Artificial Intelligence and Applications*, vol. 123. IOS Press, Amsterdam (2005)
119. Nováček, V., Smrž, P.: OLE – a new ontology learning platform. In: *RANLP-Workshop on Text Mining Research, Practice and Opportunities*, Borovets, pp. 12–16. Incoma (2005)
120. Fortuna, B., Grobelnik, M., Mladenic, D.: OntoGen: semi-automatic ontology editor. In: Smith, M.J., Salvendy, G. (eds.) *Human Interface and the Management of Information. Interacting in Information Environments*. Symposium on Human Interface held as Part of HCI International. *Lecture Notes in Computer Science*, vol. 4558, pp. 309–318. Springer, Berlin (2007)
121. Manzano-Macho, D., Gómez-Pérez, A., Borrajo, D.: Unsupervised and domain independent ontology learning: combining heterogeneous sources of evidence. In: *European Language Resources Association (ed.) Proceedings of the Sixth International Language Resources and Evaluation (LREC 2008)*, Marrakech (2008)
122. Nováček, V., Laera, L., Handschuh, S., Davis, B.: Infrastructure for dynamic knowledge integration – automated biomedical ontology extension using textual resources. *J. Biomed. Inf.* **41**(4), 816–828 (2008)
123. Gacitua, R., Sawyera, P., Rayson, P.: A flexible framework to experiment with ontology learning techniques. *Knowl.-Based Syst.* **21**(3), 192–199 (2008); 27th SGAI International Conference on Artificial Intelligence (AI 2008), Cambridge (2008)
124. Sertkaya, B.: OntoComP: a protege plugin for completing OWL ontologies. In: Aroyo, L., Traverso, P. (eds.) *Demo Proceedings of the Sixth European Semantic Web Conference (ESWC 2009)*, Heraklion. *Lecture Notes in Computer Science*, vol. 5554, pp. 898–902. Springer, Berlin (2009)
125. Simperl, E., Tempich, C., Vrandečić, D.: A methodology for ontology learning. In: Buitelaar, P., Cimiano, P. (eds.) *Ontology Learning and Population: Bridging the Gap between Text and Knowledge*. *Frontiers in Artificial Intelligence and Applications*, vol. 167, pp. 225–249. IOS Press, Amsterdam (2008)
126. Nováček, V., Dabrowski, M., Kruk, S.R., Handschuh, S.: Extending community ontology using automatically generated suggestions. In: Wilson, D., Sutcliffe, G. (eds.) *Proceedings of the 20th International Florida Artificial Intelligence Research Society Conference (FLAIRS 2007)*, Key West (2007)
127. Blomqvist, E.: OntoCase – a pattern-based ontology construction approach. In: Meersman, R., Tari, Z. (eds.) *On the Move to Meaningful Internet Systems 2007: (OTM Conferences)*. *Lecture Notes in Computer Science*, vol. 4803, pp. 971–988. Springer, Berlin (2007)
128. Völker, J., Blomqvist, E.: D3.8.2 evaluation of methods for contextualized learning of networked ontologies. Technical report 3.8.2. Institute AIFB, University of Karlsruhe, NeOn deliverable (2009)
129. Dellschaft, K., Staab, S.: On how to perform a gold standard based evaluation of ontology learning. In: Cruz, I.F., et al. (eds.) *Proceedings of the Fifth International Semantic Web Conference (ISWC 2006)*, Athens, GA. *Lecture Notes in Computer Science*, vol. 4273, pp. 228–241. Springer, Heidelberg (2006)
130. Brewster, C., Jupp, S., Luciano, J., Shotton, D., Stevens, R.D., Zhang, Z.: Issues in learning an ontology from text. *BMC Bioinform.* **10** (Suppl 5), S1 (2009)
131. Abecker, A., Bernardi, A., Hinkelmann, K., Kühn, O., Sintek, M.: Toward a technology for

- organizational memories. *IEEE Intell. Syst.* **13**(3), 40–48 (1998)
132. Schmidt, A.: Bridging the gap between knowledge management and e-learning with context-aware corporate learning solutions. In: Althoff, K.-D., Dengel, A., Bergmann, R., Nick, M., Roth-Berghofer, T. (eds.) *Proceedings of the Third Conference on Professional Knowledge Management (WM 2005)*, Kaiserslautern. *Lecture Notes in Artificial Intelligence*, vol. 3782, pp. 203–213. Springer, Berlin (2005)
 133. Stojanovic, N.: *Ontology-based information retrieval: methods and tools for cooperative query answering*. Ph.D. thesis, Universität Karlsruhe (TH) (2005)
 134. Tran, H.M., Lange, C., Chulkov, G., Schönwälder, J., Kohlhase, M.: Applying semantic techniques to search and analyze bug tracking data. *J. Netw. Syst. Manag.* **17**(3), 285–308 (2009)
 135. Bhogal, J., Macfarlane, A., Smith, P.: A review of ontology-based query expansion. *Inf. Process. Manag.* **43**(4), 866–886 (2007)
 136. Manning, C.D., Raghavan, P., Schütze, H.: *Introduction to Information Retrieval*. Cambridge University Press, New York (2008)
 137. van Rijsbergen, C.J., Crestani, F., Lalmas, M. (eds.): *Information Retrieval: Uncertainty and Logics: Advanced Models for the Representation and Retrieval of Information*. Kluwer, Norwell (1998)
 138. Meghini, C., Sebastiani, F., Straccia, U., Thanos, C.: A model of information retrieval based on a terminological logic. In: Korfhage, R., Rasmussen, E.M., Willett, P. (eds.) *SIGIR*, pp. 298–307. ACM, New York (1993)
 139. Richter, M.M., Aamodt, A.: Case-based reasoning foundations. *Knowl. Eng. Rev.* **20**(3), 203–207 (2005)
 140. Lei, Y., Motta, E., Domingue, J.: *OntoWeaver-S: supporting the design of knowledge portals*. In: Motta, E., Shadbolt, N., Stutt, A., Gibbins, N. (eds.) *Proceedings of the 14th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2004)*, Whittlebury Hall. *Lecture Notes in Computer Science*, vol. 3257, pp. 216–230. Springer, Berlin (2004)
 141. Abel, F., Henze, N.: User awareness and personalization in semantic portals. In: *Proceedings of the Fourth International Semantic Web Conference (ISWC 2005)*, Galway. *Lecture Notes in Computer Science*, vol. 3729. Springer, Berlin (2005)
 142. Mäkelä, E., Hyvönen, E., Saarela, S., Viljanen, K.: *OntoViews – a tool for creating semantic web portals*. In: McIlraith, S.A., et al. (eds.) *Proceedings of the Third International Semantic Web Conference (ISWC 2004)*, Hiroshima. *Lecture Notes in Computer Science*, vol. 3298, pp. 797–811. Springer (2004)
 143. Maedche, A., Staab, S., Stojanovic, N., Studer, R., Sure, Y.: *Semantic portal: the SEAL approach*. In: Fensel, D., Hendler, J.A., Lieberman, H., Wahlster, W. (eds.) *Spinning the Semantic Web*, pp. 317–359. MIT Press, Cambridge (2003)
 144. Lara, R., Han, S.H., Lausen, H., Stollberg, M., Ding, Y., Fensel, D.: An evaluation of semantic web portals. In: *Proceedings of the IADIS Applied Computing International Conference*, Lisbon (2004)
 145. Hildebrand, M., van Ossenbruggen, J.R., Hardman, L.: An analysis of search-based user interaction on the semantic web. Technical report. Centrum Wiskunde & Informatica, Amsterdam. <http://nwww.cwi.nl/en/system/files/u9/INS-E0706.pdf> (2007). Accessed 4 Aug 2010
 146. Huu Hoang, H., Min Tjoa, A.: The state of the art of ontology-based query systems: a comparison of existing approaches. In: *Proceedings of the IEEE International Conference on Computing and Informatics (ICOCI 2006)*, Kuala Lumpur (2006)
 147. Mangold, C.: A survey and classification of semantic search approaches. *Int. J. Metadata Semant. Ontol.* **2**(1), 23–34 (2007)
 148. Bhagdev, R., Butters, J., Chakravarthy, A., Chapman, S., Dadzie, A.-S., Greenwood, M.A., Iria, J., Ciravegna, E.: *Doris: managing document-based knowledge in large organisations via semantic web technologies*. In: Golbeck, J., Mika, P. (eds.) *Semantic Web Challenge, CEUR Workshop Proceedings*, vol. 295. CEUR-WS.org (2007)
 149. Biesalski, E., Abecker, A.: Skill-profile matching with similarity measures. In: Manolopoulos, Y., Filipe, J., Constantopoulos, P., Cordeiro, J. (eds.) *ICEIS (Selected Papers)*. *Lecture Notes in Business Information Processing*, vol. 3, pp. 210–218. Springer, Berlin (2006)

150. Hyvönen, E.: *Semantic Portals for Cultural Heritage*. Springer, Heidelberg (2009)
151. Kauppinen, T., Väättäin, J., Hyvönen, E.: Creating and using geospatial ontology time series in a semantic cultural heritage portal. In: Bechhofer, S., et al. (eds.) *Proceedings of the Fifth European Semantic Web Conference (ESWC 2008)*, Tenerife. *Lecture Notes in Computer Science*, vol. 5021, pp. 110–123. Springer, Berlin (2008)
152. Schreiber, G., Amin, A., Aroyo, L., van Assem, M., de Boer, V., Hardman, L., Hildebrand, M., Omelayenko, B., van Osenbruggen, J., Tordai, A., Wielemaker, J., Wielinga, B.: Semantic annotation and search of cultural-heritage collections: the multimediaN E-culture demonstrator. *J. Web Semant.* **6**(4), 243–249 (2008)
153. Wang, Y., Stash, N., Aroyo, L., Gorgels, P., Rutledge, L., Schreiber, G.: Recommendations based on semantically enriched museum collections. *J. Web Semant.* **6**(4), 283–290 (2008)
154. Alexopoulou, D., et al.: GoPubMed: ontology-based literature search for the life sciences. <http://rewerse.net/deliverables/m42/a2-d7.pdf> (2007). Accessed 1 Jan 2011
155. Jonquet, C., Musen, M.A., Shah, N.: A system for ontology-based annotation of biomedical data. In: Bairoch, A., Cohen Boulakia, S., Froidevaux, C. (eds.) *Proceedings of the Fifth International Workshop on Data Integration in the Life Sciences (DILS 2008)*, Evry. *Lecture Notes in Computer Science*, vol. 5109, pp. 144–152. Springer, Heidelberg (2008)
156. Suominen, O., Hyvönen, E., Viljanen, K., Hukka, E.: HealthFinland—a national semantic publishing network and portal for health information. *J. Web Semant.* **7**(4), 287–297 (2009)
157. Tzouveli, P., Schmidt, A., Schneider, M., Symvonis, A., Kollias, S.: Adaptive reading assistance for the inclusion of learners with dyslexia. In: *Proceedings of the IEEE Conference on Advanced Learning Technologies (ICALT 2008)*, Santander, Cantabria. IEEE Computer Society, Washington, DC (2008)
158. Busse, S. Kutsche, R.-D., Leser, U., Weber, H.: *Federated information systems: concepts, terminology and architectures*. Technical report, Fachbereich Informatik, TU Berlin. <http://user.cs.tu-berlin.de/~rkutsche/Publicationen/index.html> (1999). Accessed 4 Aug 2010
159. Wiederhold, G., Genesereth, M.: The conceptual basis for mediation services. *IEEE Intell. Syst.* **12**(5), 38–47 (1997)
160. Wache, H., Vögele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., Hübner, S.: *Ontology-based integration of information – a survey of existing approaches*. In: Stuckenschmidt, H. (ed.) *Workshop: Ontologies and Information Sharing (IJCAI 2001)*, Seattle (2001)
161. Cruz, I.F., Xiao, H.: Using a layered approach for interoperability on the semantic web. In: *Proceedings of the Fourth International Conference on Web Information Systems Engineering (WISE 2003)*. IEEE Computer Society
162. Doan, A., Halevy, A.Y.: Semantic integration research in the database community: a brief survey. *AI Mag.* **26**(1), 83–94 (2005)
163. Euzenat, J., Shvaiko, P.: *Ontology Matching*. Springer, Heidelberg (2007)
164. Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. *Knowl. Eng. Rev.* **18**(1), 1–31 (2003)
165. Saggion, H., Funk, A., Maynard, D., Bontcheva, K.: Ontology-based information extraction for business intelligence. In: Aberer, K., et al. (eds.) *Proceedings of the Sixth International Semantic Web Conference, Second Asian Semantic Web Conference (ISWC/ASWC 2007)*, Busan. *Lecture Notes in Computer Science*, vol. 4825, pp. 843–856. Springer, Berlin (2007)
166. Kalfoglou, Y., Hu, B., Reynolds, D., Shadbolt, N.: *Semantic integration technologies survey – 6th month deliverable of the project CROSI: capturing representing and operationalising semantic integration*. Technical report. University of Southampton, School of Electronics and Computer Science. [http://www.aktors.org/crosi/deliverables/\(2005\)](http://www.aktors.org/crosi/deliverables/(2005)). Accessed 4 Aug 2010
167. Maier, A., Schnurr, H.-P., Sure, Y.: Ontology-based information integration in the automotive industry. In: *Proceedings of the Second International Semantic Web Conference (ISWC 2003)*, Sanibel Island. *Lecture Notes in Computer Science*, vol. 2870, pp. 897–912. Springer, Berlin (2003)
168. Lu, J., Wang, S., Capretz, M.A.M.: A service oriented ontology management framework in the automotive retail domain. In: *Proceedings of the 11th IEEE International Conference on*

- Computational Science and Engineering (CSE 2008) – Workshops, Sao Paulo, pp. 239–244 (2008)
169. Vujasinovic, M., Ivezic, N., Kulvatunyou, B., Barkmeyer, E., Missikoff, M., Taglino, F., Marjanovic, Z., Miletic, I.: A semantic-mediation architecture for interoperable supply-chain applications. *Int. J. Comput. Integr. Manuf.* **22**(6), 549–561 (2008)
 170. Barillot, C., Benali, H., Dameron, O., Dojat, M., Gaignard, A., Gibaud, B., Kinkingnehun, S., Matsumoto, J.-P., Pelegriani-Issac, M., Simon, E., Temal, L., Valabregue, R.: Federating distributed and heterogeneous information sources in neuroimaging: the NeuroBase project. Technical report 1712. <http://www.irisa.fr/visages/demo/Neurobase/Download/IRISAResearchReport1712.pdf> (2005). Accessed 4 Aug 2010
 171. Angele, J., Gesmann, M.: Semantic information integration with software AGs information integrator. In: Proceedings of the Second International Conference on Rules and Rule Markup Languages for the Semantic Web (RuleML 2006), Athens, GA. <http://2006.ruleml.org/> (2006). Accessed 4 Aug 2010
 172. Angele, J., Gesmann, M.: Integration of customer information using the semantic web. In: Cardoso, J., et al. (eds.) *The Semantic Web: Real-World Applications from Industry. Semantic Web And Beyond Computing for Human Experience*, vol. 6, pp. 191–208. Springer, New York (2007)
 173. Bizer, C., Heath, T., Berners-Lee, T.: Linked data – the story so far. *Int. J. Semant. Web Inf. Syst.* **5**(3), 1–22 (2009)
 174. Coetzee, P., Heath, T., Motta, E.: SparqPlug: generating linked data from legacy HTML, SPARQL and the DOM. In: Proceedings of the First International Workshop on Linked Data on the Web (LDOW 2008) at 17th International World Wide Web Conference (WWW 2008), Beijing. <http://events.linkedata.org/ldow2008/papers/05-coetzee-heath-sparqplug.pdf> (2008). Accessed 4 Aug 2010
 175. Becker, C., Bizer, C.: DBpedia mobile – a location-aware semantic web client. In: Proceedings of the Semantic Web Challenge at Seventh International Semantic Web Conference (ISWC 2008), Karlsruhe. *Lecture Notes in Computer Science*, vol. 5318. Springer, Berlin (2008)
 176. Das Sarma, A., Dong, X., Halevy, A.Y.: Bootstrapping pay-as-you-go data integration systems. In: Tsong-Li Wang, J. (ed.) *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2008)*, Vancouver, pp. 861–874. ACM, New York (2008)
 177. Kobilarov, G., Scott, T., Raimond, Y., Oliver, S., Sizemore, C., Smethurst, M., Lee, R.: Media meets semantic web – how the BBC uses DBpedia and linked data to make connections. In: Proceedings of the Sixth European Semantic Web Conference (ESWC 2009), Heraklion. *Lecture Notes in Computer Science*, vol. 5554. pp. 723–737. Springer, Berlin (2009)
 178. Jackson, P.: *Introduction to Expert Systems*, 3rd edn. Addison-Wesley, Boston (1998)
 179. Payne, V.L., Metzler, D.P.: Hospital Care Watch (HCW): an ontology and rule-based intelligent patient management assistant. In: Proceedings of the 18th International Symposium on Computer-Based Medical Systems (CBMS 2005), Dublin, pp. 479–484. IEEE Computer Society, Washington, DC (2005)
 180. Lee, C.-S., Wang, M.-H.: Ontology-based intelligent healthcare agent and its application to respiratory waveform recognition. *Expert Syst. Appl.* **33**(3), 606–619 (2007)
 181. Lee, C.-S., Wang, M.-H.: Ontological fuzzy agent for electrocardiogram application. *Expert Syst. Appl.* **35**(3), 1223–1236 (2008)
 182. Bobillo, F., Delgado, M., Gmez-Romero, J.: Representation of context-dependant knowledge in ontologies: a model and an application. *Expert Syst. Appl.* **35**(4), 1899–1908 (2008)
 183. Angele, J., Mönch, E., Nierlich, A., Oppermann, H., Rudat, H., Schnurr, H.-P.: Applications and good practices of semantic technologies. <http://www.eddie-moench.de/publications> (2006). Accessed 4 Aug 2010
 184. Britton, C.: *IT Architectures and Middleware: Strategies for Building Large. Integrated Systems*. Addison-Wesley, Boston (2001)
 185. Taylor, I.J., Harrison, A.: *From P2P and Grids to Services on the Web-Evolving Distributed Communities*, 2nd edn. Springer, London (2009)
 186. Kuroпка, D., Tröger, P., Staab, S., Weske, M. (eds.): *Semantic Service Provisioning*. Springer, Berlin (2008)
 187. McIlraith, S.A., Son, T.C., Zeng, H.: Semantic web services. *IEEE Intell. Syst.* **16**(2), 46–53 (2001)

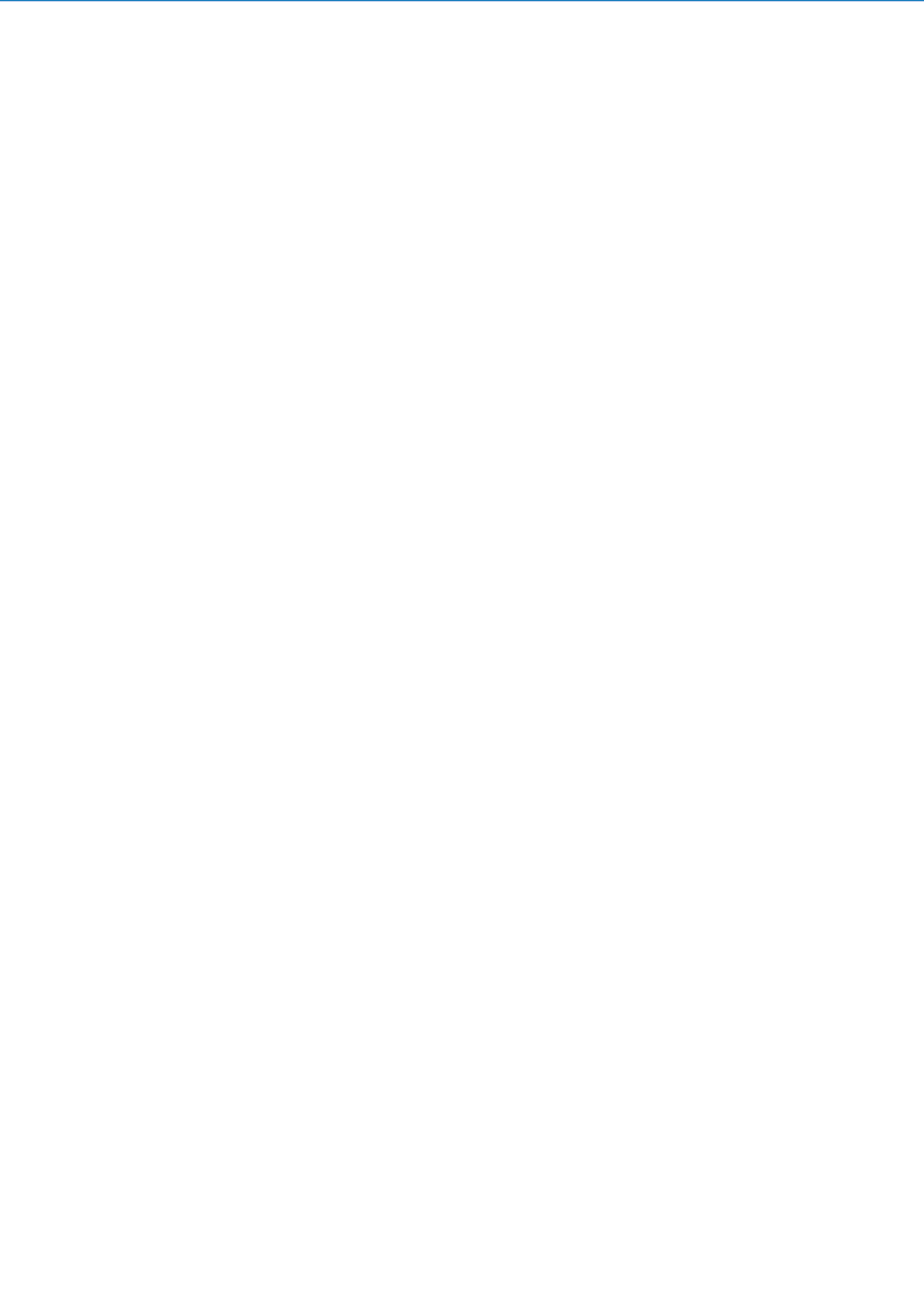
188. Di Martino, B.: Semantic web services discovery based on structural ontology matching. *Int. J. Web Grid Serv.* 5(1), 46–65 (2009)
189. Keller, U., Lara Hernandez, R., Fensel, D., Kifer, M., Polleres, A., Lausen, H., Zhao, C.: A logical framework for web service discovery. In: *Proceedings of the ISWC 2004 Workshop: Semantic Web Services: Preparing to Meet the World of Business Applications*, Hiroshima. CEUR Online Proceedings, CEUR-WS.org (2004). Accessed 23 Dec 2010
190. Kourtis, D., Paraskakis, I.: Combining SAWSDL, OWL-DL and UDDI for semantically enhanced web service discovery. In: *Proceedings of the Fifth European Semantic Web Conference (ESWC 2008)*, Tenerife. *Lecture Notes in Computer Science*, vol. 5021, pp. 614–628. Springer, Berlin (2008)
191. Klusch, M., Fries, B., Sycara, K.: Automated semantic web service discovery with OWLS-MX. In: *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006)*, Hakodate, pp. 915–922. ACM, New York (2006)
192. Klusch, M., Kaufer, F.: WSMO-MX: a hybrid semantic web service matchmaker. *Web Intell. Agent Syst.* 7(1), 23–42 (2009)
193. Srivastava, B., Koehler, J.: Web service composition – current solutions and open problems. In: *Workshop on Planning for Web Services at the 13th International Conference on Automated Planning & Scheduling (ICAPS 2003)*, Trento, pp. 28–35 (2003). <http://www.isi.edu/integration/workshops/icaps2003-p4ws/>. Accessed 23 Dec 2010
194. Hakimpour, F., Sell, D., Cabral, L., Domingue, J., Motta, E.: Semantic Web service composition in IRS-III: the structured approach. In: *Proceedings of the Seventh IEEE International Conference on E-Commerce Technology (CEC 2005)*, Munich, pp. 484–487. IEEE Computer Society (2005)
195. Rao, J., Küngas, P., Matskin, M.: Composition of semantic web services using linear logic theorem proving. *Inf. Syst.* 31(4), 340–360 (2006)
196. Lecue, F., Delteil, A., Leger, A.: Applying abduction in semantic web service composition. In: *Proceedings of the IEEE International Conference on Web Services (ICWS-2007)*, Salt Lake City, pp. 94–101 (2007)
197. Lécué, F., Léger, A.: A formal model for semantic web service composition. In: *Proceedings of the Fifth International Semantic Web Conference (ISWC 2006)*, Athens, GA. *Lecture Notes in Computer Science*, vol. 4273, pp. 385–398. Springer, Heidelberg (2006)
198. Lécué, F., Léger, A., Delteil, A.: DL reasoning and AI planning for web service composition. In: *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT 2008)*, Sydney, pp. 445–453 (2008)
199. Kim, J., Spraragen, M., Gil, Y.: An intelligent assistant for interactive workflow composition. In: *Proceedings of the Ninth International Conference on Intelligent User Interfaces (IUI 2004)*, Funchal, Madeira, pp. 125–131. ACM, New York (2004)
200. Zheng, G., Bouguettaya, A.: Discovering pathways of service oriented biological processes. In: *Proceedings of the Ninth International Conference on Web Information Systems Engineering (WISE 2008)*, Auckland. *Lecture Notes in Computer Science*, vol. 5175, pp. 189–205. Springer, Berlin (2008)
201. Foster, I., Kesselman, C. (eds.): *The GRID: Blueprint for a New Computing Infrastructure*, 2nd edn. Morgan Kaufmann, San Francisco (2004)
202. De Roure, D., Jennings, N.R., Shadbolt, N.R.: The semantic grid: past, present, and future. *Proc. IEEE* 93(3), 669–681 (2005). <http://eprints.ecs.soton.ac.uk/9976/>. Accessed 4 Aug 2010
203. Schollmeier, R.: A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. In: *Proceedings of the Peer-to-Peer Computing (P2P) Conference*, pp. 101–102. IEEE Computer Society, Washington, DC (2001)
204. Garcia-Molina, H., Crespo, A.: Semantic overlay networks for P2P systems. Technical report 2003-75, Stanford InfoLab. <http://ilpubs.stanford.edu:8090/627/> (2003). Accessed 4 Aug 2010
205. Gu, T., Pung, H.K., Zhang, D.: Information retrieval in schema-based P2P systems using

- one-dimensional semantic space. *Comput. Netw.* **51**(16), 4543–4560 (2007)
206. Gu, T., Pung, H.K., Zhang, D.: A semantic P2P framework for building context-aware applications in multiple smart spaces. In: Kuo, T.-W., Hsing-Mean Sha, E., Guo, M., Tianruo Yang, L., Shao, Z. (eds.) *EUC. Lecture Notes in Computer Science*, vol. 4808, pp. 553–564. Springer, Berlin (2007)
 207. Aberer, K., Cudre-Mauroux, P., Hauswirth, M., van Pelt, T.: GridVine: building internet-scale semantic overlay networks. In: McIlraith, S.A., et al. (eds.) *Proceedings of the Third International Semantic Web Conference (ISWC 2004)*, Hiroshima. *Lecture Notes in Computer Science*, vol. 3298, pp. 107–121. Springer, Berlin (2004)
 208. Haase, P., Broekstra, J., Ehrig, M., Menken, M., Mika, P., Olko, M., Plechawski, M., Pyszlak, P., Schnizler, B., Siebes, R., Staab, S., Tempich, C.: Bibster – a semantics-based bibliographic peer-to-peer system. In: McIlraith, S.A., et al. (eds.) *Proceedings of the Third International Semantic Web Conference (ISWC 2004)*, Hiroshima. *Lecture Notes in Computer Science*, vol. 3298, pp. 122–136. Springer, Berlin (2004)
 209. Vaquero, L.M., Rodero-Merino, L., Caceres, J., Lindner, M.: A break in the clouds: towards a cloud definition. *SIGCOMM Comput. Commun. Rev.* **39**(1), 50–55 (2009)
 210. Youseff, L., Butrico, M., da Silva, D.: Towards a unified ontology of cloud computing. In: *Grid Computing Environments Workshop (GCE08)*, held in conjunction with *ACM/IEEE SuperComputing Conference (SC 2008)*, Austin (2008)
 211. Mika, P., Tummarello, G.: Web semantics in the clouds. *IEEE Intell. Syst.* **23**(5), 82–87 (2008)
 212. Mentzas, G., Friesen, A. (eds.): *Semantic Enterprise Application Integration for Business Processes: Service-Oriented Frameworks*. IGI Global, Hersley (2009)
 213. Haniewicz, K., Kaczmarek, M., Zyskowski, D.: Semantic web services applications – a reality check. *Wirtschaftsinformatik* **50**(1), 39–46 (2008)
 214. Klusch, M., Zhing, X.: Deployed semantic services for the common user of the web: a reality check. In: *Proceedings of the International Conference on Semantic Computing (ICSC 2008)*, Santa Clara, pp. 347–353. IEEE Computer Society, Washington, DC (2008)
 215. Roman, D., Klien, E.: SWING – a semantic framework for geospatial services. In: Scharl, A., Tochtermann, K. (eds.) *The Geospatial Web: How Geo-Browsers, Social Software and the Web 2.0 are Shaping the Network Society*. Springer, London (2007)
 216. Zaharia, R., Vasiliu, L., Hoffman, J., Klien, E.: Semantic execution meets geospatial web services: a pilot application. *Trans. GIS* **12**(1), 59–73 (2009)
 217. Dogac, A., Laleci, G.B., Kirbas, S., Kabak, Y., Sinir, S.S., Yildiz, A., Gurcan, Y.: Artemis: deploying semantically enriched web services in the healthcare domain. *Inf. Syst.* **31**(4), 321–339 (2006)
 218. Schumacher, M., Helin, H., Schuldt, H. (eds.): *CASCOM: Intelligent Service Coordination in the Semantic Web*. Birkhäuser, Basel (2008)
 219. della Valle, E., Zhao, G., Monteleone, G., Cerizza, D., Celino, I., Estublier, J., Vega, G., Kerrigan, M., Ramirez, J., Villazon-Terrazas, B., Guarrera, P.: SEEMP: a semantic interoperability infrastructure for e-Government services in the employment sector. In: *Proceedings of the Fourth European Semantic Web Conference (ESWC 2007)*, Innsbruck. *Lecture Notes in Computer Science*, vol. 4519, pp. 220–234. Springer, Berlin (2007)
 220. Gugliotta, A., Domingue, J., Cabral, L., Tanasescu, V., Galizia, S., Davies, R., Gutiérrez-Villarias, L., Rowlett, M., Richardson, M., Stincic, S.: Deploying semantic web services-based applications in the e-Government domain. *J. Data Semant.* **10**, 96–132 (2008)
 221. Milanovic, M., Gasevic, D., Kaviani, N.: Ontologies for software engineering. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*, 2nd edn. Springer, Dordrecht (2009)
 222. Tetlow, P., Pan, J.Z., Oberle, D., Wallace, E., Uschold, M., Kendall, E.: Ontology driven architectures and potential uses of the semantic web in systems and software engineering, W3C Working Draft, Working Group Note 2006/02/11, W3C. <http://www.w3.org/2001/sw/BestPractices/SE/ODA/> (2006). Accessed 4 Aug 2010
 223. Happel, H.-J., Seedorf, S.: Applications of ontologies in software engineering. In: *Proceedings of the Workshop on Semantic Web Enabled*

- Software Engineering (SWESE 2006) at International Semantic Web Conference (ISWC 2006), Athens, GA (2006)
224. Kiko, K., Atkinson, C.: Integrating enterprise information representation languages. In: Proceedings of the International Workshop on Vocabularies, Ontologies and Rules for The Enterprise (VORTE 2005), Enschede, pp. 41–50 (2005)
225. Happel, H.-J., Korthaus, A., Seedorf, S., Tomczyk, P.: KOntoR: an ontology-enabled approach to software reuse. In: Zhang, K., Spanoudakis, G., Visaggio, G. (eds.) SEKE-2006, pp. 349–354 (2006)
226. Ankolekar, A., Sycara, K., Herbsleb, J., Kraut, R., Welty, C.: Supporting online problem-solving communities with the semantic web. In: Proceedings of the 15th International Conference on World Wide Web (WWW 2006), Edinburgh, pp. 575–584. ACM, New York (2006)
227. Dang, Q.V., Bac, C., Berger, O., Dao, X.S.: Improving community awareness in software forges by semantical aggregation of tools feeds. http://libresoft.es/oldsite/Activities/Research_activities/WoPDaSD2008_files/Paper4.pdf (2008). Accessed 4 Aug 2010
228. Doerr, M.: The CIDOC conceptual reference module: an ontological approach to semantic interoperability of metadata. *AI Mag.* **24**(3), 75–92 (2003)
229. Sinclair, P.A.S., Goodall, S., Lewis, P.H., Martinez, K., Addis, M.J.: Concept browsing for multimedia retrieval in the SCULPTEUR project. In: Proceedings of the Workshop on Multimedia and the Semantic Web at Second European Semantic Web Conference (ESWC 2005), Heraklion. Lecture Notes in Computer Science, vol. 3532. Springer, Berlin. <http://eprints.ecs.soton.ac.uk/10913/> (2005). Accessed 4 Aug 2010
230. Vrochidis, S., Doulaverakis, C., Gounaris, A., Nidelkou, E., Makris, L., Kompatsiaris, I.: A hybrid ontology and visual-based retrieval model for cultural heritage multimedia collections. *Int. J. Metadata Semant. Ontol.* **3**(3), 167–182 (2008)
231. Hatala, M., Wakkary, R., Kalantari, L.: Rules and ontologies in support of real-time ubiquitous application. *J. Web Semant.* **3**(1), 5–22 (2005)
232. Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) AAAI Spring Symposium Semantic Web Meets E-government. Technical Report SS-06-06. AAAI Press, Menlo Park (2006)
233. Charalabidis, Y., Metaxiotis, K.: Chapter XIII: ontology-based management of e-Government knowledge. In: Rahman, H. (ed.) Social and Political Implications of Data Mining: Knowledge Management in E-Government. IGI Global, Hershey (2008)
234. Crichton, C., Davies, J., Gibbons, J., Harris, S., Shukla, A.: Semantic frameworks for e-Government. In: Proceedings of the First International Conference on Theory and Practice of Electronic Governance (ICEGOV 2007), pp. 30–39. ACM (2007)
235. Stojanovic, L.: Ontology-based change management in an eGovernment application scenario. In: Studer, R., Grimm, S., Abecker, A. (eds.) Semantic Web Services: Concepts, Technologies, and Applications. Springer, Berlin (2008)
236. Ojo, A., Janowski, T., Estevez, E.: Semantic interoperability architecture for electronic government. In: Proceedings of the 10th Annual International Conference on Digital Government Research (DG.O 2009), Puebla, pp. 63–72. Digital Government Society of North America (2009)
237. Apostolou, D., Stojanovic, N., Anicic, D.: Responsive knowledge management for public administration: an event-driven approach. *IEEE Intell. Syst.* **24**(5), 20–30 (2009). Special Issue on Transforming E-government & E-participation
238. Sidoroff, T., Hyvönen, E.: Semantic e-government portals – a case study. In: Proceedings of the Workshop on Semantic Web Case Studies and Best Practices for eBusiness (SWCASE 2005) at Fourth International Semantic Web Conference (ISWC 2005), Galway (2005)
239. Trochidis, I., Tambouris, E., Tarabanis, K.A.: An ontology for modeling life-events. In: IEEE International Conference on Services Computing (SCC 2007), Salt Lake City, pp. 719–720. IEEE Computer Society, Washington, DC (2007)
240. Stevens, R., Goble, C.A., Bechhofer, S.: Ontology-based knowledge representation for bioinformatics. *Brief. Bioinform.* **1**(4), 398–414 (2000)

241. The Gene Ontology Consortium: Creating the gene ontology resource: design and implementation. *Genome Res.* **11**(8), 1425–1433 (2001)
242. Smith, B., Ashburner, M., Rosse, C., Bard, J., Bug, W., Ceusters, W., Goldberg, L.J., Eilbeck, K., Ireland, A., Mungall, C.J., Leontis, N., Rocca-Serra, P., Ruttenberg, A., Sansone, S.-A., Scheuermann, R.H., Shah, N., Whetzel, P.L., Lewis, S.: The OBO foundry: coordinated evolution of ontologies to support biomedical data integration. *Nat. Biotechnol.* **25**(11), 1251–1255 (2007)
243. Kashyap, V., Cheung, K.-H., Doherty, D., Samwald, M., Marshall, M.S., Luciano, J., Stephens, S., Herman, I., Hookway, R.: Ontology-based data integration for biomedical research. In: Cardoso, J., et al. (eds.) *The Semantic Web: Real-World Applications from Industry. Semantic Web and Beyond Computing for Human Experience*, vol. 6, pp. 97–122. Springer, New York (2007)
244. Dietze, H., Schroeder, M.: GoWeb: a semantic search engine for the life Science Web. In: Burger, A., Paschke, A., Romano, P., Splendiani, A. (eds.) *Workshop on Semantic Web Applications and Tools for Life Sciences (SWAT4LS 2008)*, Edinburgh. <http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-435/> (2008). Accessed 4 Aug 2010
245. Bechhofer, S., Yesilada, Y., Stevens, R., Jupp, S., Horan, B.: Using ontologies and vocabularies for dynamic linking. *IEEE Internet Comput.* **12**(3), 32–39 (2008)
246. Rector, A.L., Qamar, R., Marley, T.: Binding ontologies and coding systems to electronic health records and messages. *J. Appl. Ontol.* **4**(1), 51–69 (2009)
247. Sheth, A.P., Agrawal, S., Lathem, J., Oldham, N., Wingate, H., Yadav, P., Gallagher, K.: Active semantic electronic medical records. In: Cardoso, J., et al. (eds.) *The Semantic Web: Real-World Applications from Industry. Semantic Web And Beyond Computing for Human Experience*, vol. 6, pp. 123–140. Springer, New York (2007)
248. Staab, S., Studer, R. (eds.): *Handbook on Ontologies. International Handbooks on Information Systems*, 2nd edn. Springer, Heidelberg (2009)
249. Hitzler, P., Krötzsch, M., Rudolph, S.: *Foundations of Semantic Web Technologies*. Chapman & Hall/CRC Press, Boca Raton (2009)
250. Allemang, D., Hendler, J.: *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL*. Morgan Kaufmann, San Francisco (2008)
251. d’Aquin, M., Castro, A.G., Lange, C., Viljanen, K. (eds.) *Proceedings of the First International Workshop on Ontology Repository and Editors for the Semantic Web (ORES 2010) at Seventh Extended Semantic Web Conference (ESWC 2010)*, Heraklion. Online Proceedings: <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-596/>(2010). Accessed 10 Aug 2010
252. Buitelaar, P., Eigner, T.: Evaluating ontology search. In: Garcia-Castro, R., Vrandečić, D., Gómez-Pérez, A., Sure, Y., Huang, Z. (eds.) *Proceedings of the Fifth International Workshop on Evaluation of Ontologies and Ontology-based Tools (EON 2007)*, co-located with Sixth International Semantic Web Conference (ISWC 2007), Busan. *CEUR Workshop Proceedings*, vol. 329, pp. 11–20. CEUR-WS.org (2007). Accessed 1 Jan 2011
253. Rubin, D.L., Moreira, D.A., Kanjamala, P.P., Musen, M.A.: BioPortal: a web portal to biomedical ontologies. In: Sleeman, D., Musen, M. (eds.) *AAAI 2008 Spring Symposium on Symbiotic Relationships between Semantic Web and Knowledge Engineering*. Stanford University, Palo Alto (2008)
254. Goble, C.A., Corcho, O., Alper, P., de Roure, D.: e-Science and the Semantic Web: a symbiotic relationship. In: *Proceedings of the 17th International Conference on Algorithmic Learning Theory (ALT 2006)*, Barcelona (2006)
255. Codina, J., Pianta, E., Vrochidis, S., Papadopoulos, S.: Integration of semantic, metadata and image search engines with a text search engine for patent retrieval. In: Bloehdorn, S., Grobelnik, M., Mika, P., Tran, D.T. (eds.) *Workshop on Semantic Search at Fifth European Semantic Web Conference 2008 (ESWC 2008)*, Tenerife. *Lecture Notes in Computer Science*, vol. 5021, pp. 14–28. Springer, Berlin (2008)
256. van Elst, L., Abecker, A.: *Ontologies for information management: balancing formality,*

- stability, and sharing scope. *Expert Syst. Appl.* **23**(4), 357–366 (2002)
257. Hepp, M.: Possible ontologies: how reality constrains the development of relevant ontologies. *IEEE Internet Comput.* **11**(1), 90–96 (2007)
258. Zacharias, V., Braun, S., Schmidt, A.: Social semantic bookmarking with SOBOLEO. In: Murugesan, S. (ed.) *Handbook of Research on Web 2.0, 3.0 and X.0: Technologies, Business, and Social Applications*. IGI Global, Hershey (2009)
259. Wang, H., Liu, Q., Penin, T., Fu, L., Zhang, L., Thanh Tran, D., Yu, Y., Pan, Y.: Semplore: a scalable IR approach to search the web of data. *J. Web Semant.* **7**(3), 177–188 (2009)
260. Thanh Tran, D., Haase, P., Studer, R.: Semantic search – using graph-structured semantic models for supporting the search process. In: *Proceedings of the 17th International Conference on Conceptual Structures (ICCS 2009)*, Moscow (2009)
261. Chen, H., Finin, T., Joshi, A.: An ontology for context-aware pervasive computing environments. *Knowl. Eng. Rev.* **18**(3), 197–207 (2003)
262. Gu, T., Wang, X.H., Pung, H.K., Zhang, D.Q.: An ontology-based context model in intelligent environments. In: *Proceedings of the 2004 Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2004)*, San Diego, pp. 270–275 (2004)
263. Corcho, O., Hauswirth, M., Koubarakis, M. (eds.) *Proceedings of the International Workshop on the Semantic Sensor Web (SemSensWeb 2009)*, Collocated with ESWC 2009, Crete. *CEUR Online Proceedings*. <http://www.ceur-ws.org/Vol-468> (2009). Accessed 4 Aug 2010
264. Sheth, A., Henson, C., Sahoo, S.S.: Semantic sensor web. *IEEE Internet Comput.* **12**(4), 78–83 (2008)
265. Cardoso, J., Hepp, M., Lytras, M.D. (eds.): *The Semantic Web: Real-World Applications from Industry. Semantic Web and Beyond Computing for Human Experience*, vol. 6. Springer, New York (2007)
266. Cruz, I.F., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.): *Proceedings of the Fifth International Semantic Web Conference (ISWC 2006)*, Athens, GA. *Lecture Notes in Computer Science*, vol. 4273. Springer, Heidelberg (2006)
267. McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.): *Proceedings of the Third International Semantic Web Conference (ISWC 2004)*, Hiroshima. *Lecture Notes in Computer Science*, vol. 3298. Springer, Heidelberg (2004)



14 Future Trends

Lyndon Nixon¹ · Raphael Volz² · Fabio Ciravegna³ · Rudi Studer^{4,5}

¹Semantic Technology Institute (STI) International, Vienna, Austria

²Volz Innovation GmbH, Loffenau, Germany

³University of Sheffield, Sheffield, United Kingdom

⁴FZI Research Center for Information Technology, Karlsruhe, Germany

⁵Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

14.1	<i>Introduction</i>	583
14.2	<i>Scientific and Technical Overview: Charting the Future on the Basis of the Past</i>	583
14.3	<i>Example Application and Technology Trends in Semantics</i>	586
14.3.1	Applications	586
14.3.1.1	General (Enterprise)	586
14.3.1.2	Energy	588
14.3.1.3	Production	588
14.3.1.4	Media, Art, and Entertainment	589
14.3.1.5	Health	589
14.3.1.6	Smart Life	590
14.3.1.7	Urban Computing	591
14.3.2	Other Technologies	592
14.3.2.1	Social Technology	592
14.3.2.2	3D Technology	593
14.3.2.3	Mobile Technology	593
14.3.2.4	Virtual Worlds	594
14.3.2.5	Sensor Networks	595
14.3.2.6	IP Television	595
14.3.2.7	Security/Privacy	596
14.3.3	Semantic Technologies	597
14.3.3.1	Annotation	597
14.3.3.2	Context	599
14.3.3.3	Knowledge Discovery and Data Mining (KDD)	599
14.3.3.4	Modeling	600

14.3.3.5	Reasoning	601
14.4	<i>Future Trends in Semantic Technology: Experts' Opinion</i>	602
14.4.1	Trends in Semantic Technologies by 2014	602
14.4.1.1	Individual Visions of the Invited Experts	602
14.4.2	Trends in Semantic Technologies by 2019	604
14.4.2.1	Individual Visions of the Invited Experts	604
14.4.3	Trends in Semantic Technologies by 2024	606
14.4.4	Points Raised in an Open Discussion, Inspired from the Proposed Trends	606
14.5	<i>Related Resources Including Key Papers</i>	607
14.5.1	Semantic Roadmaps	607
14.6	<i>Predicting the Next 15 Years of Technology</i>	608
14.7	<i>Roadmap for Semantic Technology to 2024</i>	611
14.8	<i>Conclusion: A Brave New World</i>	614
14.9	<i>Cross-References</i>	616

Abstract: This volume has introduced the foundations and technologies which make up the Semantic Web. Mostly the discussion has been on the state of the art, but what developments can be expected next in semantic technologies? What social and technological trends will spur and enable the next generation of semantic technology? Which application areas can one expect to gain most added value from implementation of semantic technologies in the next 15 years? With expert eyes on the crystal ball, this final chapter of the first volume will outline these future trends.

14.1 Introduction

Looking back over the past 15 years, few would have predicted the impact of the Internet in daily life. Internet technologies have effected a major transformation and spurred innovations in many other technologies and sciences. This transformation has been achieved by tearing apart prior frictions in communication and information exchange. Semantic technologies are on the cusp of becoming mainstream technologies and promise to remove many of the remaining frictions in communication and information exchange.

In this chapter, the following questions will be raised and an attempt to make an initial answer will be made, drawing on the vision and insights of experts from the semantic technologies community:

- Which set of major transformations can be expected from semantic technologies?
- What is the role of semantic technology in responding to large societal challenges?
- Which new application areas could emerge from the fusion of semantics with other technologies and sciences?
- Which technologies come next?

14.2 Scientific and Technical Overview: Charting the Future on the Basis of the Past

Looking into the past underlines why a common vision for the future of semantic technologies is beneficial: the tire-tracks of [Fig. 14.1](#) underscores how much industry builds on government-funded university research, sometimes through long incubation periods. [Fig. 14.1](#) also illustrates the interdependencies of research advances in various subfields: A complex research ecology with several concurrent advances is at work, and multiple subfields (in particular within Computer Science but also extending into other fields) are mutually reinforcing, stimulating, and enabling one another.

One of the most important messages is the long, unpredictable incubation period which requires steady work and funding between initial exploration and commercial deployment. To secure continued funding, a common shared vision is essential. This vision needs to evolve and capture real-world requirements that can change quickly.

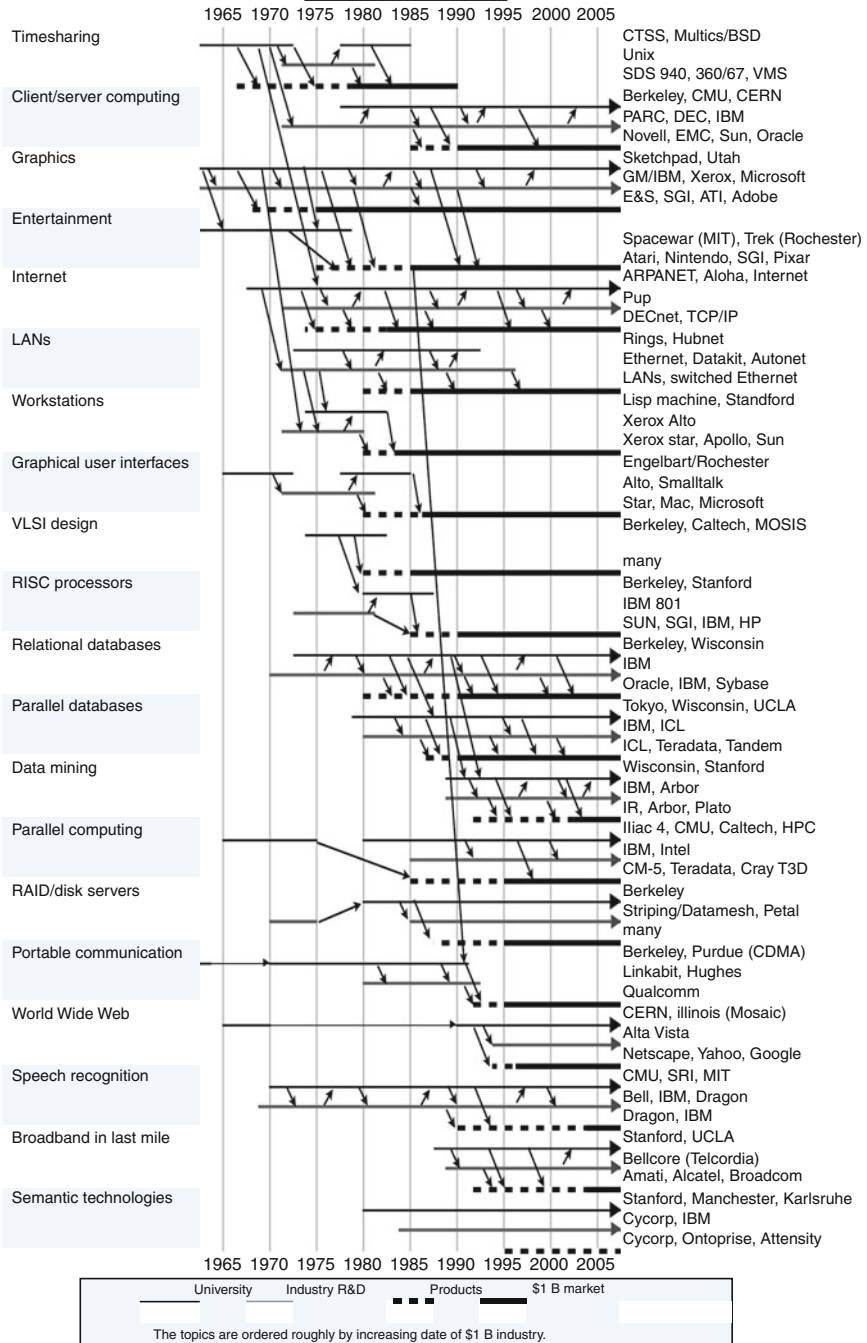


Fig. 14.1

From research to \$1 B market (Augmentation of chart by Computer Science and Telecommunications Board, National Research Council of Canada [1])

Semantic technologies can look back to more than 25 years of university and industry R&D, early research on semantic networks being summarized by Allen and Frisch in 1982. Cyc can be regarded as the earliest commercial product sold by Cycorp since 1995 after more than a decade of R&D (<http://www.cyc.com>). Since then, the commercial product offer has increased drastically in recent years. There have been mergers and acquisitions (such as IBM's acquisition of Unicorn) culminating in the merger of US-based Attensity with German-based *empolis* and *living-E* that has created the largest semantic technology vendor to date. However, the long envisioned \$1 billion market has not yet appeared.

How do others envision the future of semantic technology? In their 2008 annual Hype Cycle for Web and User Interaction, Gartner describes the Semantic Web as being over 10 years away from mainstream adoption, yet at least coming up out of the *trough of disillusionment* and into the *slope of enlightenment* [2] – as reality has set in, away from the utopian visions of AI and intelligent Web agents, and into a more bottom-up enriching of the Web with more structured data, drawing from the lessons of Web 2.0. The full mainstreaming of semantics on the Web and in the enterprise will be “multiple evolutionary steps” according to Gartner's report entitled “Finding and Exploiting Value of Semantic Technologies in the Web” (2007). Here, they predict 80% of public Web pages having some semantic markup by 2012, but only 15% will use ontologies (i.e., most markups will serve to provide controlled vocabularies and structure, but not logic-based reasoning). In a chart entitled the Semantic Web Evolution, they predict that semantic data in terms of full RDF/OWL will be mainstream on the Web by 2017, and the Semantic Web itself may first become a reality (as a “semantic environment” on the Web) by 2027. Finally, and in line with the other reports, Gartner's “Priority Matrix for Web and User Interaction Technologies 2008” places Semantic Web in the “high benefit” category with “more than 10 years” to mainstream adoption.

So both lessons from the past and visions of the future expect a slow, but steady, establishment of semantic technologies in the mainstream IT and Internet infrastructure. It may be summarized that the mainstreaming (broad presence and use) of semantics is estimated (according to the opinion of analysts such as Gartner) at:

- ca. 2012 for semantic markup on the Web without much use of ontologies and critical mass of semantic applications in early adopter enterprises
- ca. 2017 for semantic markup on the Web with use of ontologies and breakthrough of enterprise semantic applications into the mainstream
- ca. 2018–2024 for wide application of semantic technology into (increasingly if not fully Web-based) enterprise computing as a “standard” approach, like RDBMS today
- ca. 2025–2027 for the socio-technological tipping point, where semantics become a pervasive reality in a ubiquitous Web (Gartner calls this the “semantic environment”)

In the remainder of the chapter, opinions of semantic technology experts and an overview of the specific literature (relating to individual application and technology areas) will be brought together to chart in more detail the future trends and eventual uptake of semantic technology in all areas of future (digital) business and society.


14.3 Example Application and Technology Trends in Semantics

In a survey of individual visions for semantic technologies in the next 15 years among invited semantic technology experts, a wide array of visions was collected for future applications of semantic technologies. All applications identified assumed the omnipresence of basic Internet technologies and the Web. Almost all visions presented assumed that core semantic technologies, such as semantic annotation, knowledge extraction, search, modeling, and reasoning, will interact with other emerging technologies to enable new applications and solve real-world problems. The following technologies were identified as particularly interesting candidates for a pairing with semantic technologies:

- Social technologies (such as social networks)
- 3D technologies (e.g., virtual worlds)
- Mobile networks
- Sensors and sensor networks (e.g., RFID)
- New media technologies (such as IPTV)

In their interaction with these other technologies, semantic technologies are considered to reach the mainstream in the next years in many application areas:

- In enterprises where they will increase the effectiveness of administration and management (e.g., IT asset management)
- To manage energy more intelligently
- To increase flexibility and effectiveness of production
- To enable new ways of interaction in media, art, and entertainment
- To improve health care
- To improve urban life, for example, by optimizing traffic flow and other types of urban computing

The interplay between application areas, semantic technologies, and non-semantic technologies, leading to new technological and social developments in the next 5–15 years, is illustrated in  Fig. 14.2.

For each application and technology, a short overview can be given of the envisioned trends as follows.

14.3.1 Applications

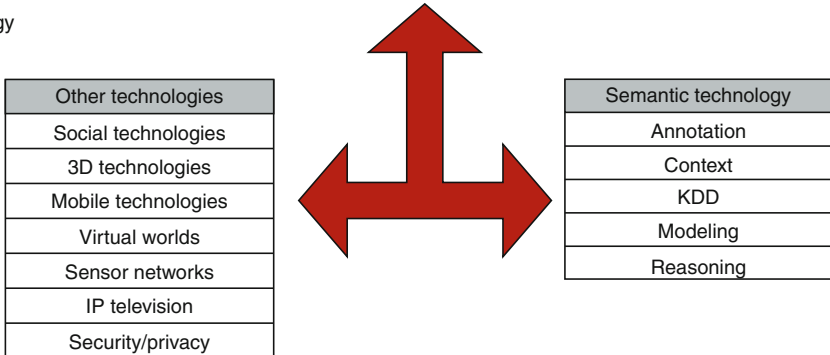
14.3.1.1 General (Enterprise)

Business use of semantic technologies is considered on the verge of widespread application, seen in both the W3C business use-case collections and attendance figures at the main business conferences for semantic technologies: SemTech with >1,000 attendees in 2009 (<http://www.semantic-conference.com/>), European Semantic Technologies

Application areas

Enterprise						
			Personal			
				Societal		
General	Energy	Production	Media, art, and entertainment	Health care	Smart life	Urban computing

Technology



Examples

Urban computing applies stream reasoning using data from sensor networks to guide traffic flow

■ Fig. 14.2

Future mainstream application areas for semantic technologies

Conference with >200 attendees in 2008 (<http://www.estc2008.com>). The W3C in October 2009 places semantic technologies at the end of the early adopters phase, shortly before mainstream markets (early majority) [3] (based on the Chasm Group’s Technology Adoption Life Cycle [4], where a chasm exists between the early adopters phase and the early majority, so future mainstream uptake presupposes crossing that chasm). “Semantic web technology is on the verge of becoming commercially viable for businesses looking to develop their web capabilities” (according to John Davies, Head of Semantic Technology at British Telecom), and “there will be significant increases in the real-world application of semantic technology over the next 12–18 months” [5]. Given that the interview was at end of 2008, this would place the tipping point for mainstream semantic application usage in enterprises to the end of 2009/beginning of 2010. Looking beyond this to widespread adoption, the Semantic Wave 2008 report from Mills Davis forecasts [6]:

- Public and private sector R&D relating to semantic technologies in the 2008–2010 period will exceed \$8 billion.
- Global ICT markets for semantic technology–infused products and services will grow from \$2.1 billion in 2006 to \$52.4 billion in 2010.

- Enterprise adoption of semantic technologies will increase dramatically. Public and private sector enterprises represent three-fourths of global ICT spending.

The reality may be rather less spectacular. It is more likely that early adopters will continue to lead the way in increasingly more critical activities being supported by mature semantic tools and technologies. The reality of wide usage of semantics in business processes will be somewhere between then and a decade later, which is Gartner's more conservative bet.

14.3.1.2 Energy

The energy sector is seen as one of the key areas for early adoption of semantic technologies. Chevron's input to the W3C Semantic Web Education and Outreach working group noted several uses of semantics in the oil and gas industry [7]. The industry has some of the aspects which act as key business drivers for semantics: large amounts of heterogeneous data being generated daily from multiple sources from which information value must be extracted; information search and access across the data sources and formats; information needing to be standardized and integrated across systems. Semantics in the energy sector is a trend which will continue, especially as technologies mature and gain acceptance such that they become increasingly applied in the core, critical business processes. In particular, scalability of semantic tools will be a critical technology barrier to its breakthrough into mainstream energy IT systems.

14.3.1.3 Production

Manufacturing and production chains involve very complex processes and rules [8]. When the manufacturing industry started using IT, there was less attention paid to heterogeneity as systems operated more independently and large investments were made in large and complex systems; today's IT systems for manufacturing on the other hand must – if the manufacturer is to remain competitive – find ways to seamlessly integrate different manufacturing subsystems with one another and also with the systems of the other players in the production chain (suppliers, purchasers, etc.). If semantic technology can enable that seamless integration in the production sector, this will act as a major driver for uptake as it leads to improved collaboration, reduced integration costs, and increased business agility. Key challenges to be overcome in the application of semantic technology are:

- Achieving consensus on meaning across organizations
- Many overlapping standards for manufacturing interoperability lacking explicit and rigid definitions of terms
- Globalization, making it vital that information sharing is done correctly, efficiently, and inexpensively
- People, meaning that it is still people who provide the knowledge needed to interpret information and make decisions based on their tacit understanding of it

Here, semantic SOA (Web Services) could become the critical technology driver, enabling a semantic interoperability layer to be applied on top of the existing and emerging SOA being introduced into enterprise IT infrastructure. (SOA is seen as already a mainstream technology, and Gartner foresees 80% of mission-critical enterprise IT processes as being SOA-based in 2010.)

14.3.1.4 Media, Art, and Entertainment

The media sector faces new challenges in the scale and complexity of media being produced and shared. Online media in particular needs improved retrieval, adaptation, and presentation if content owners are to win market share in a broad and overfilled market. Semantic media involves providing media objects with a semantic description (annotation) and using this to offer better search for media, automated adaptation (personalization, contextualization), and meaningful presentation. Creating the media annotations is the biggest challenge, since full automation of the process is difficult and lacks precision, while manual annotation is time consuming and cannot keep up with the scale of media being produced. A few media organizations have begun to lead the way in using and demonstrating semantics, for example, the BBC has begun to publish its online content with RDF.

The arts, that is, cultural heritage, is another sector in which semantics are gaining traction. Museums, for example, have large amounts of metadata about their collections which cannot be easily interpreted or reused due to non-digital, non-semantic, and proprietary approaches. Again, some pioneers such as the Rijksmuseum in Amsterdam are taking the first steps to digitize and annotate their collections and explore the new possibilities which are realized [9].

Semantics can provide future solutions to the media, arts, and entertainment sector for their problems with large scales of heterogeneous non-textual content, and for the emerging challenges in realizing attractive and competitive content offers on a ubiquitous Web with millions of content channels. The cost of creating the semantic data tends to be larger at present than the benefits purchased from its creation, so while the potential benefit from semantics will continue to grow as Web media becomes more ubiquitous (making having a Unique Selling Point ever more critical for a content owner and provider), the actual costs of semantics must still fall through better, more automated content annotation tools and approaches.

The semantic multimedia community will achieve important steps in standardization and maturity of technology and tools in the short to medium term. So a semantic technology breakthrough in the media sector can be expected in the medium to long term.

14.3.1.5 Health

eHealth is a leading and exemplary adopter of semantic technologies. It demonstrates an increasing focus on end users, allowing semantics to be applied in real and important

problems and to improve health-care standards to the benefit of society and individuals. Health care fits well for semantic technology adoption because it works with complex, yet structured, and well-defined terminologies and models, where experimentation (e.g., drug development) and application (e.g., medical care) both need to capture significant amounts of heterogeneous data and derive from them significant trends and conclusions.

AW3C Health Care and Life Sciences group exists to explore the “use of Semantic Web technologies . . . to improve collaboration, research and development, and innovation adoption.” It has over 60 participants from 40 organizations, including Agfa, AstraZeneca, Cleveland Clinic, Eli Lilly, HL7, Merck, and Pfizer. Task forces explore, for example, the application of semantics in areas such as drug safety and efficacy, adaptable clinical protocols and pathways, and clinical observations interoperability.

As a significant prerequisite for semantic eHealth, the community is particularly active in generating health-care and life science ontologies. These are already being applied in pharmaceutical and biological research departments. The use of ontologies allows for the exploration and discovery of new knowledge out of large knowledge bases (e.g., results of experiments). Documents and resources can be found at the W3C group blog (<http://www.w3.org/blog/hcls>).

While semantic eHealth is a reality now, and indicators exist of actual usage of semantics in pharmaceutical and life science R&D, grander visions of the semantic health-care community may still require a longer-term focus. For example, the semantic HEALTH project which was tasked with a roadmap for “semantic interoperability for better health and safer healthcare” [10] noted that interoperability of Electronic Health Record Systems may take another 20 years to achieve.

14.3.1.6 Smart Life

Smart living refers to the ubiquity of the Web and Web-enabled devices in an individual’s daily life, so that every personal and social activity may be supported in a meaningful and beneficial manner. It covers intelligent homes as well as smart cars and personal devices (e.g., Internet-aware clothing).

Home networks are an emerging trend, slowed in recent times by lack of interoperable standards across all devices and the need to reflect the expectations of consumers. Solutions are emerging through common standards (overseen by the Digital Living Network Alliance) and clearer delineation of devices. Device interoperability and computational awareness of the home environment support vital “assistive living” scenarios. The business driver for semantics is found in the current state of the art, where various technologies are needed to be integrated in the smart home, each of which provides a fragment of the necessary functionality. A vision for adaptive, personalized, and context-aware smart homes needs a future infrastructure which can support the full richness of the smart home vision. Semantics are seen to have a role in defining knowledge about the environment in a reusable and processable manner (e.g., captured through sensors), integrating services and data between devices and enabling the definition of rules for

reactive situations in smart home environments (e.g., analyzing the mood of the person and adjusting the home lighting or background music accordingly). An architecture to enable this “Semantic Smart Home” [11] has been presented. The building blocks exist but the acceptance and uptake by the industry is notably slower than in other sectors which have been previously mentioned. In line with the general trend toward more mainstream uptake of semantics across industry, the Semantic Smart Home may be realized in the mid-term. More complex reasoning cases which foresee the dynamic composability of Semantic Web Services can be seen as having a longer-term solution.

Cars are another emerging target for the integration of IT and telecommunications technology to allow for in-car intelligent services, typically related to vehicle location (through GPS) and the surroundings. This field is called *telematics*. European car manufacturers are very active in developing telematic solutions for their automobiles, especially in response to increasing competitiveness in the global market from low-cost car manufacturers. Telematic solutions (e.g., intelligent in-car navigation systems) are already in development and deployment in the automobile industry [12]. Semantics can play a role in integrating data and composing services to deliver functionality to the end user. However, there is a long way foreseen for new technologies which potentially change the driving experience to move out of the research lab and into the roads; even the potential integration of social networks into driving is cautioned as being “a long way off” (Venkatesh Prasad, Ford USA quoted in Telematics Update, Nov 5 2009).

Finally, the introduction of semantic technologies to Internet-aware personal objects lead to the ideas of “smart clothing,” “smart glasses,” etc., where normal, everyday objects will be enhanced by electronics, Internet accessibility, and the ability to provide services to the user (e.g., clothes changing color or design to adapt to the surrounding situation, glasses overlaying information about objects seen by the wearer). This requires an interdisciplinary approach within the fashion, textile, electronics, and IT industries, as well as preparing the general public with respect to acceptance of potentially disruptive ideas.

14.3.1.7 Urban Computing

Urban computing refers to the increasing ubiquity of the Internet and devices connected to the Internet in the urban environment leading to smart cities. Current technology lacks the capability of effectively solving urban computing problems as it requires combining a huge amount of static knowledge about the city with larger, real-time data being generated by sensors and devices in a heterogeneous and noisy manner. To act upon the data acquired, their combination is insufficient; rather there must be intelligent reasoning to draw in-time inferences.

Current projects provide test solutions in more controlled and controllable environments, but the technological challenge remains to satisfactorily deal with large-scale, heterogeneous, and “dirty” data. Hence, current research seeks new solutions based on semantics [13], yet mature solutions for true, open urban environments may be expected first in the middle to long term.

14.3.2 Other Technologies

14.3.2.1 Social Technology

Social technology reflects the increasingly person-centric nature of Web technologies (often collected under the term “Web 2.0”) where content is generated by the users and the major focus of the technology is to enable the contact between users over the content platform (e.g., blogs, wikis, Facebook, Twitter). It has enjoyed phenomenal uptake on the computer-based Web, and will be expected to become an even more fundamental part of human activity through the increasing ubiquity of the Web in mobile, home, and personal devices.

Social Web technology has enjoyed a very quick uptake and success rate based on simplicity of use, low barrier to entry, and, needless to say, the human factor. As such, it has proved a useful challenge to the classical top-down approach of the Semantic Web, whose strong requirement on formal logical correctness, the need to learn how to develop ontologies (knowledge modeling), and focus on machines over humans has arguably proven a significant barrier to wider uptake. This has resulted in two approaches to semantics and the social Web:

- Social *Semantic* Web seeks to build upon social technology and improve aspects such as search and retrieval, or the interlinking of people and content, through the use of semantics.
- *Social* Semantic Web seeks to explore how Semantic Web data can be created more easily based on social software principles.

The former covers combination of Web 2.0 and semantic technologies into new tools such as semantic wikis, semantic blogs, semantic social networks (e.g., <http://twine.com>), and semantic activity streams (e.g., <http://identi.ca>).

The latter includes building lightweight ontologies and annotations out of tags and folksonomies, wiki-based ontology engineering, and games for semantic content creation (e.g., [14], <http://ontowiki.net>, <http://ontogame.sti2.at>).

Such hybrid approaches are at the beginning of wider usage, and certainly do not yet challenge the classical tools and technologies in the domain, whether it is wikis and blogs on one hand, or ontology editors and annotation tools on the other. Yet the combination of social and semantic technology has a clear mutual benefit, since some knowledge may be derived from the sheer scale of human-generated data combined with statistical methods (so-called collective intelligence) and yet other knowledge can only be explicitly pre-provided in the form of ontologies and semantic models (the “implicit knowledge” from which humans have to be able to derive conclusions, and must be given explicitly to machines for reasoning. For example, if I go to Paris to see the Piney Woods, and you go to Paris to see the can-can dance, we may conclude I went to Paris, Texas, and you went to Paris, France. A machine may only be able to conclude this if it has the implicit knowledge that the Piney Woods are in Texas and the can-can dance comes from France.) Their combination is foreseen by visionaries such as Nova Spivack as being the basis for their complete breakthrough as part of an emerging

digital existence (always online, always producing and consuming data). The resulting “MetaWeb” [15] benefits from the increased information connectivity enabled by semantics and the increased social connectivity enabled by social software to result in a connection of intelligence. Driven by these natural benefits, social and semantic technology should draw ever closer over the next decade, and the following decade (2020–2030) should, according to this vision at least, see the resulting intelligence-connection precursing what Spivack calls the “global brain.” Details on the current state of the relationship between semantics and social technology can be found in [Social Semantic Web](#).

14.3.2.2 3D Technology

The third dimension has always been a part of human perception, but in the digital world it has had a shorter existence. Today, on the other hand, computers are capable of rendering highly complex 3D scenes which can even be mistaken for real by the human eye. 3DTV is on the cusp of market introduction. A new IT segment must deal with the capturing of 3D objects, their manipulation and management, in application domains from health care to cultural heritage.

Challenges in the 3D technology domain include how to describe 3D objects for their indexing, storage, retrieval, and alteration. Semantics provide a means to improve the description, search, and reuse of complex 3D digital objects. Awareness of the value and potential use of this technology in the 3D media community is at an early stage [16]. It is being promoted to industry through initiatives like FOCUS K3D (<http://www.focusk3d.eu>) which has application working groups for the domains of medicine and bioinformatics, gaming and simulation, product modeling, and archeology. A survey on the state of the art in cultural heritage [17] notes that progress is being made on standardized metadata schemes and ontologies, current limitations relate to methodologies and the lack of specialized tools for 3D knowledge management, yet this could be addressed in the short to medium term.

14.3.2.3 Mobile Technology

Mobile devices are projected to grow to 4.39 billion units in 2011. By 2012 the number of connected mobile devices may equal the human population of the world. The smartphone market grew by 15% between 2008 and 2009. Mobile applications, increasingly sophisticated as devices show greater computing capacity, are seen as a major future market: the iPhone application store, which has over 85,000 applications and which were downloaded over 2 billion times in the first year, is now being joined by Android, Palm, Nokia, and others. Hence, three key trends can be seen which will be impacting the mobile technology domain in the near term:

- Growth of the number of (Internet capable) mobile devices in use
- Increasing computing power of those devices
- Resulting usage of mobile-based services and applications

Falling connection costs and greater availability of wireless LAN networks and flat-rate Internet mobile access are the market drivers for the latter point, which is seen as a significant enabler for the Internet of Services.

The challenges of providing personalization and contextualization (e.g., location-based) to application use in a mobile scenario, dynamicism and adaptation to the data/service provision, and building the necessary scalability and flexibility into mobile networks have driven the consideration of semantic solutions. Telecommunications operators have been one of the early adopters of the technology, both in research and development and, in some cases, actual products (Vodafone live! Portal, <http://www.w3.org/2001/sw/sweo/public/UseCases/Vodafone/>). They are also a strong area for wider uptake of the semantic technologies, often based around service architectures [18].

14.3.2.4 Virtual Worlds

Virtual worlds have been around as an idea since virtual reality in the 1980s; however, their breakthrough into the social mainstream may have arguably been initiated by the Web-based computer application Second Life. Started in 2003, there are almost 19 million registered users as of April 2010 (taken from <http://secondlife.com/xmlhttp/secondlife.php>). Since Second Life popularized the idea of virtual worlds, others have established themselves more strongly around shared themes and goals, such as fantasy role-playing (Runescape, World of Warcraft), kids games (Club Penguin, Webkinz), or building up virtual farms, zoos, cafes, etc. (the Facebook app Farmville has 35 million monthly users).

Virtual worlds are a natural extension of 3D technology into reflecting the perceptive realities of this world and have also found applicative usage in domains such as medicine, social analysis, education, and eCommerce. Making virtual worlds “react” more realistically to actions and activities performed by the actors of that world requires a (semantic) understanding of the objects rendered in the world and the events which (can) occur between them. There is also a trend to more closely couple real and virtual worlds through (real-world) sensors which generate data streams to cause the virtual world to reflect the real in near-real time. This leads to a requirement to handle increasing scales of heterogeneous, dirty data for information extraction and actionable inference within the virtual world.

As in the 3D technology field, barriers to the use of semantic technologies lie in the need to agree on the vocabularies and schema for description of the worlds (which now goes beyond the form of objects, and encapsulates what can be done with them, how they react to external events, etc.), as well as the availability of appropriate tools for the creation and maintenance of semantic virtual worlds. Hence, it is likely that semantics will first need to experience wide uptake in 3D technology systems before it also further develops into a technology for virtual worlds in the medium to long term.

Projects such as *Semantic Reality* (<http://www.semanticreality.org>) provide exciting longer-term visions of a virtual world tightly connected to the real world, with trillions of sensors able to ensure a close correlation between both. This requires an even greater

research effort, integrating the domains of sensor networks, embedded systems, ambient intelligence, networking, distributed systems, social networking, and the Semantic Web. Even within the latter, further work is required on provisioning large-scale, open semantic infrastructures, semantically enriched social networks, semantic sensor networks (see the next item), emergent semantics, real-time query processing, and aspects of integrity, confidentiality, reputation, and privacy [19].

14.3.2.5 Sensor Networks

The Future Internet will increasingly involve machine-generated data rather than user-generated, sourced in Internet-connected sensors, RFID tags, and other embedded devices. An enormous amount of (potentially useful and reusable) data are being generated and processed by applications in the fields of environment, agriculture, health, transportation, surveillance, and public security, among others. In the future, this could explode with 50–100 thousand billion connected objects (“Internet of Things”).

“Semantic Sensor Web” has established itself as a term for the annotation of sensor data with spatial, temporal, and thematic semantic metadata. This is a reaction to the need to better extract actionable intelligence from increasing scales of heterogeneous and often dirty data coming from sensor networks. Semantics also facilitate the integration and communication between the networks. Leading projects in the field are already able to demonstrate real prototypes built on top of existing and emerging sensor data standards [20].

The extension of sensor network deployments to use semantic technologies requires (1) appropriate modeling and language support for describing objects, (2) reasoning over data generated by objects, (3) semantic execution environments and architectures that accommodate network requirements, and last but not least (4) scalable storing and communication infrastructure [21]. Initial rollouts of semantic sensor networks can be expected in the short to medium term, extending existing standards and approaches with semantics, while larger-scale, critical sensor applications will develop longer term in line with both the technological emergence of the Internet of Things and parallel maturing of semantic technology in terms of scalability, heterogeneity, and handling inconsistent or incomplete data.

14.3.2.6 IP Television

IP Television refers to the convergence of Internet and Television, which is also happening outside of the television set (e.g., also Web-based TV, Mobile TV). Currently it is focused on new types of services around television such as EPGs, programming on demand, and live TV pause. An emerging trend in IPTV is toward Web integration through widgets, which are lightweight self-contained content items that make use of open Web standards (HTML, JavaScript) and the back-channel of the STB to communicate with the Web (typically in an asynchronous manner). Yahoo! and Intel, for example, presented their Widget Channel at the CES in January 2009, where Web content such as Yahoo! news and

weather, or Flickr photos, could be displayed in on-screen widgets on TV. Sony and Samsung will go to market in 2010 with Internet-enabled televisions. A 2009 survey found that there should be a gradual but steady uptake of TV Internet usage with “the mass market inflection point occurring over the next 3–5 years” (http://oregan.net/press_releases.php?article=2009-01-07).

Parallel to this, research into semantic IPTV applications and solutions is being established in academic and industry labs. A key focus for semantics is the formal description of the programming and user interests to provide for a better personalization of the TV experience (EU project NoTube, <http://www.notube.tv>, 2009–2012) as well as a formal description of networks and content to enable a better delivery of complex services (myMedia, <http://www.semantic-iptv.de>).

A major barrier to uptake by broadcasters and content providers is the lack of support for semantic technology in the legacy broadcast systems. Shifts in the provider-side IT infrastructure to Internet-based (even cloud-based) infrastructures should give an opening for an introduction of semantics into the production systems of the television and media companies. Vocabularies and technologies will need to converge on specific standards to encourage industry acceptance, which should emerge in this next “uptake” period. As Internet-TV reaches the mass market point (possibly by 2014), companies will seek Unique Selling Points for their products and services, which will drive the incorporation of semantic technologies into IPTV infrastructures and packages. More information on the current use of semantics in IPTV can be found in [▶ Multimedia, Broadcasting, and eCulture](#).

14.3.2.7 Security/Privacy

In a study on critical issues in the Future Internet conducted by RAND Europe for the Netherlands Ministry of Economic Affairs [22], *identity*, *privacy*, and *trust* were all indicated as being highly important by all experts participating in the study. Identity refers to the ability to refer unambiguously to an actor in the Internet (human or device) and privacy reflects as well the need to be able to maintain and protect one’s own identity. The problem is compounded by the rapid increase of the number of devices connecting to the Internet as well as their differing roles and purposes.

As noted in [23], semantic technologies have a role to play in solving challenges relating to identity and privacy:

- URIs/URLs are a big success story on how identity could be handled in large, open distributed environments and could be applied in an identity management solution in the Future Internet.
- The “Identity Anarchy” which is mainly due to heterogeneity of models, devices, applications, and languages could benefit from the mediation and interoperability research carried out within the Semantic Web.
- Semantic technologies have/will provide ontological models for various identity-related aspects such as policies, profiles, networking, etc.

Existing frameworks for privacy, trust, and security management do not take semantic technology into consideration. However, models for these frameworks need to be able to represent the actors in a transaction, how they know one another, what they know about one another, other knowledge about them, previous experience, etc. The lack of models which are sufficiently descriptive, dynamic, and reusable across applications and domains will drive the uptake of semantic technology parallel to social trends which will drive the necessity of such solutions, such as the increased use of the Internet for transactions and increased number of actors in a transaction, not all of whom may be or can be knowable. Significant research work [24] has considered trust and trust relationship models using semantics. These are affinal to reasoning and rules, to build security and trust policies that can be checked and modified, and evolve in dynamic situations. Further work is needed on application of semantic security and trust models in real-world situations, such as in Service Level Agreements (SLA).

14.3.3 Semantic Technologies

14.3.3.1 Annotation

Annotation is a vital prerequisite for the use of semantic technology. The vast majority of data being produced by tools and services are not semantically annotated, leading to the necessity to extract semantic descriptions either in advance or on-the-fly from legacy data before they can be used in semantics-based systems.

Semantic annotations can be divided between automatic and manual approaches, as well as text-based and non-text-based approaches.

Manual, text-based approaches supported by appropriate tools to select the correct concept (class, instance, property) for a selected text snippet are quite mature and usable by nonexperts. The cost in terms of time taken to produce the annotations has led to approaches which (semi-) automatically annotate text documents based on natural language processing (NLP) and text mining techniques. These tend to provide the best results when used with sufficiently large text corpora and address clear, narrow domains (e.g., medical documents, which use consistent and agreed terms). The advantages of machine learning when applied to text documents at scale have been applied in providing web-based text annotation tools (e.g., OpenCalais from Thomson Reuters at www.opencalais.com) capable of identifying named entities and annotating text which may cover much wider, more general domains. While a subject of ongoing research, accuracy is already very satisfactory and provides the possibility of the semantic annotation of text to the mainstream market, which should drive use of semantic technology in the short term by lowering the barrier to acquiring semantics from text-based data.

Multimedia annotation faces a greater challenge in that the data to be annotated are of a non-textual form, for example, aural (audio), visual (image), or audio-visual (video). Here, given that text-based annotation approaches are more developed, one approach is to determine text relating to the non-textual media to be able to apply a text-based

annotation approach. For example, speech recognition, association to a subtitles track or script, and extraction of the associated textual description from metadata or a web page can be used with aural media. With visual media, computer vision techniques are applicable to attempt to identify objects and events within image/video (potentially guided by extractable, related text in order to restrict identification to particular domains). The mapping of low-level audio-visual characteristics to high-level semantic descriptions are referred to as “bridging the semantic gap” in media. The accuracy of resulting media annotations is proportionate to the amount of manual oversight involved in the annotation process, as humans continue to be better than machines at identifying objects and events in media outside of controlled, narrow domain situations (e.g., a soccer match). While sub-domains such as video object tracking or face detection are already well developed, true computer vision close or even the same as biological (human) vision in terms of identifying (arbitrary) objects, events, or even abstract concepts (in arbitrary situations) is seen in similar terms to “Strong AI” (machines becoming as capable as humans in undertaking tasks). While many scientists doubt the possibility, others see computing power approaching that of the human brain in the future (technology futurist Ray Kurzweil would put the timepoint in which computers are capable of the same feats as the human brain to 2015–2025 [25]).

Semantic annotation tools are already quite mature. A report in 2006 [26] identified seven requirements for annotation tools, and concluded that many were already fulfilled or on their way, noting the following as still needing to be addressed in particular:

- Support for trust, provenance, and access policies
- Support for ontology maintenance
- Support for the evolution of the (annotated) documents
- Usability for nonexpert knowledge workers

Possibly the most effective short-to-medium-term solution for semantic annotation is to ensure that knowledge about and from the data is captured as implicitly as possible at the time of creation through the used tool, since any attempt to determine that implicit knowledge subsequently without recourse to involving the human expert is always subject to possible error. Even non-textual data may be more effectively annotated by these means as the actual context of the media capture is available to the annotation system and ambiguities can be addressed by simple requests to the user which will prove much less effort-intensive than a later attempt at annotation, potentially by someone other than the media producer, without recourse to the full context of the time of the media creation. This requires a better integration of annotation approaches into existing data creation and capture tools as opposed to separate, specialized annotation frameworks.

Long term, the maturity of technologies and techniques, tested on huge scales of data, combined with sufficient computing power may become such that the Holy Grail of fully automated annotation becomes possible. An overview of the current state of semantic annotation can be found in [▶ Semantic Annotations and Retrieval: Manual, Semiautomatic, and Automatic Generation](#).

14.3.3.2 Context

Context refers, in its general sense, to “that which surrounds and gives meaning to something else” (Free Online Dictionary of Computing, www.foldoc.org). In semantic technologies, it refers to the need to capture (possibly in a formal manner) the circumstances in which knowledge is being obtained, modeled, changed, used, or acted upon, in order to enable a semantic system to make use of that information when interacting with other users, systems, or data, that is, for purposes of personalization, adaptation, filtering, ranking, or selection (or, more generally described, to enable “contextualization”).

Capturing context for semantic applications is currently limited to specific aspects which are easier to identify and model, for example, location expressed in degrees of longitude and latitude. More complex context models for fine-grained contextualization in semantic applications are missing. Research considers in particular mobile and pervasive environments where the contextual information can play a particularly significant role for appropriate data and service provision.

Incorporation of semantic approaches to context modeling and contextualization of data and services on the basis of a semantic context model will follow from the convergence of two medium-term trends: the incorporation of semantic technologies into computing environments – particularly those where context plays a significant role such as mobile and pervasive – and the increased use in the meantime of non-semantic approaches to contextualization (e.g., GPS data for location-based services).

14.3.3.3 Knowledge Discovery and Data Mining (KDD)

KDD refers to the act of extracting useful knowledge from data making use of analytic, statistical, and mining techniques. The persistent and rapid growth of data to be processed in IT systems due to the Internet, increased digitalization of content and processes, and trends such as sensor networks have created an immense need for KDD methodologies. With respect to semantic technology, KDD can aid in extracting useful knowledge for ontology modeling (cf. the next point) as well as ontology population (cf. i) (annotation above).

Current challenges in KDD include the need to handle dynamic data streams (e.g., from sensors), processing in real time, extracting actionable knowledge from text (computational linguistics), and handling data which are not clean, consistent, or correct (probabilistic approaches) [27]. Breakthroughs in KDD could contribute significantly to the challenges of ontology creation and instance data creation, particularly when dealing with large-scale data (an increasingly relevant issue in both enterprise and Web environments). However, areas such as computational linguistics have not shown much progress in decades, and it is clear that new approaches need to be taken.

To the extent that the computational understanding of data may be a question of computational power, as noted above, a breakthrough may be expected in the period 2015–2025 if computers achieve capabilities close to those of the human brain (which includes the understanding of natural language).

14.3.3.4 Modeling

Modeling refers to the tools and techniques making it possible for knowledge experts to capture their knowledge of a domain in the form of a formal logic-based model (an ontology). This encompasses means to formalize knowledge, build an ontology, evaluate and refine it, as well as to maintain and evolve it.

The task of modeling knowledge requires both experts in the domain to be modeled and experts in formally modeling that knowledge in an ontology. As this is usually not found in the same person, ontology modeling tends to be a collaborative process. Better means to collaboratively model ontologies have arisen with the growth of Web 2.0 techniques (e.g., wikis) but are still relatively immature. New challenges arise in formal knowledge capture in a relatively informal community-based approach, and lightweight ontologies (which model the domain but have few axioms or rules) are becoming seen as a more feasible means to promote wider semantic technology uptake by lowering the barrier to ontology creation (this will exist in parallel to a continued approach to create rigorously formal ontologies for the domains in which this is a necessity, for example, for health and life science applications).

Formal ontology engineering has taken methodologies inspired from the software and knowledge engineering domains, and many methodologies for knowledge modeling have been proposed, from Cyc (1990) to NeOn (2006). A 2009 study of ontology engineering practice [28] found that “it has already become an established discipline with respect to the maturity and level of acceptance of its main components, methodologies, etc. whereas further research should target economic aspects. . . and the customization of existing technology to the specifics of vertical domains.” Ontology editors are largely mature (in their commercial variants) and in the surveyed practices, ontology evaluation/refinement has tended to be carried out by expert judgment rather than according to any methodology. As ontologies begin to be established in the IT processes of companies and industry sectors, it is the maintenance and evolution aspects which will represent the most significant challenge – while methodologies and tools have existed for many years, complete solutions are needed which integrate into an ontology management software suite and support the domain experts (e.g., automatic change management). These aspects should come to the fore of research and application in the short term in anticipation of wider ontology usage in real-world situations by the medium term.

While the lightweight ontology trend will bring more semantic data to broader use in the short term (consider the Linked Open Data meme where structured semantic data are published on the Web using lightweight RDF structures – www.linkeddata.org, or “folksonomies” – taxonomies which emerge out of tagging trends in Web 2.0), it raises new challenges for the future Semantic Web before such models can be effectively reused for validation, data integration, or inference tasks, for example, in enterprise situations. These include [29]:

- Methodologies for community-driven modeling
- Automatic refactoring of lightweight semantic models

- Design patterns
- “Softer” mapping relations between ontologies
- How lightweight a semantic model can get in order to still be useful
- Semantic model visualization

Both formal and informal approaches to modeling will continue to exist in parallel and there will be use cases exclusively for one or the other, for example, the former (formal) approach will remain the choice of critical business applications where rigor and correctness of inference are vital, while the latter (informal) will gain traction in more open, public-facing, Web-based applications (e.g., semantically enhanced versions of Web 2.0 sites). An overview of the current status of semantic modelling can be found in [▶ Ontologies and the Semantic Web](#).

14.3.3.5 Reasoning

Reasoning refers to the means to infer new knowledge from the basis of instance semantic data, ontologies, and a formal logic model on which those ontologies are based (i.e., base axioms). Reasoners form a critical part of the semantic application architecture, whenever inference or semantic validation is required (as opposed to simply using RDF as a flexible data model).

Reasoners exist for many different ontology languages/flavors of logic, from RDFS and OWL Lite to theorem provers working in first-order logic. The “classical” Semantic Web approach uses a decidable subset of first-order logic called Description Logic. Several reasoners exist for RDFS and OWL(-DL plus/minus some constructs) level reasoning and are achieving maturity as well as scalability up to billions of RDF triples – providing the usual trade-offs between data size, reasoning complexity, and completeness/correctness requirements. Open challenges focus on providing flexible reasoning frameworks that can adapt to the needs of differing inference tasks, as well as a better handling of inconsistency, fuzziness, and approximation in data inference.

While reasoning over finitely large datasets in controlled environments (e.g., on an enterprise network) is mature enough for commercial application today (using tractable logic fragments such as OWL-DL), large-scale distributed reasoning tasks (the ultimate example of which being reasoning over the Web) currently face scalability limits which can be gradually overcome by relaxing the inference guarantees of completeness and correctness, as well as permitting greater approximation and fuzziness in the calculation. This is the subject of a major European research effort (LarKC) which is aiming to enable reasoning over 20 billion RDF triples with completeness and 100 billion RDF triples with incompleteness by 2010 [30]. An overview of the OWL can be found in [▶ KR and Reasoning on the Semantic Web: OWL](#) and of web-scale Reasoning in [▶ KR and Reasoning on the Semantic Web: Web-scale Reasoning](#).

14.4 Future Trends in Semantic Technology: Experts' Opinion

Having considered the general view on these areas, the results of the exercise in collecting experts' individual visions for semantic technologies are summarized in the following in terms of the general consensus for semantic technologies in the next 5, 10, and then 15 years.

14.4.1 Trends in Semantic Technologies by 2014

While experts in the semantics community foresee the first mature semantic technologies in the next years, there was the recognition that the maturity of particular aspects of semantic technology by 2014 did not necessarily mean their immediate uptake and integration in the application areas seen as particularly standing to benefit from them. Rather, the next years would see the maturity and adoption of semantic technology from research labs to early adopters with particular urgent need for semantics (e.g., eHealth), and the “widespread deployment” would be mostly across those sectors (research labs, public sector, health, telecommunications) which are already leading in semantic technology application. In these end scenarios, semantics can achieve their final (industrial) maturity and act as convincing showcases to the wider community. In parallel, the application areas given in [Fig. 14.2](#) will develop from being today's “trends” to being part of the digital mainstream. In doing so, they will reach their natural limits (without semantic technology).

14.4.1.1 Individual Visions of the Invited Experts

Diego Berrueta, Researcher, CTIC Foundation

- ▶ Recognizing that the current Internet still has a lot of problems (spam, trust, precise search, system interoperability) the first steps must be to provide solutions here. Our hot research topics for the next years are: context and profiling, data grounding, knowledge sharing and management, and pervasive sensor networks.

Michael Brodie, Chief Scientist, Verizon Communications

- ▶ In 5 years from now the world will be even more networked as before and the Internet even more ubiquitous. We will be measuring data and traffic in zettabytes (each representing 250 billion DVDs). We should find niches in emerging technologies where we can bring semantics: such examples are social networks/computing, master data management, business rule management. The Gartner hype curves for different technology trends don't mention semantics at all! The main thing is to find problems where semantic technologies provide a unique advantage and develop real solutions.

Fabio Ciravegna, Professor, University of Sheffield

- ▶ “Semantic technologies will gain traction for large scale distributed organizations. A future trend with high impact will be the “Corporate Semantic Web” (see Gartner 2006).”

Marko Grobelnik, Researcher, J Stefan Institute

- ▶ The role of semantics can be to reduce and control the complexity of information, so semantic technologies are really a horizontal area technology for anywhere that has to deal with abstractions. We expect semantic research to aid a defragmentation of scientific discipline, that semantics will aid machine understanding of text and become very practical at the “long tail” of lightweight reasoning, sensors will make real time solutions on data streams more important and “cloud computing” will address the scalability of heavyweight reasoning.

Dunja Mladenic, Researcher, J Stefan Institute

- ▶ Semantics in 2014 will address all data modalities, including media types, enterprise networks, personal content and community. Knowledge generation and flow between actors in the networked world will be modelled by semantic technologies, enabling the understanding and prediction of interactions and incorporating knowledge bases and reasoning to produce more observable concrete data from content and interactions. There will be more gadgets, more services, more fun, and ... less privacy.

Lyndon Nixon, Senior Researcher, STI International

- ▶ Media is no longer tied to a single device, but will be as ubiquitous as the Internet. Added value services will be part of media delivery and media storage will be in the cloud. Semantics will be beginning to enter the mainstream in 5 years, and we will be on the cusp of rich, intelligent added value media services.

Rudi Studer, Professor, Institute AIFB/Karlsruhe Institute of Technology

- ▶ The trends are towards more mobile (PDA) and ubiquitous computing, the emergence of the Internet of Services and even greater information overload. Semantic technologies will become more mature and more pragmatic, with the market mainly moving towards lightweight applications, and the main usage being data integration across service providers and contextualised service/information delivery and consumption. Semantics will be for the masses: emerging as a byproduct of daily work and collaboration. New application areas emerging from the fusion of semantics and other technologies include intelligent assist[an]ce in automobiles and augmented reality. Tackling global problems, semantics may help resource-aware and sustainable IT, better information access for the “digitally disadvantaged,” and eHealth.

Emanuele Della Valle, Assistant Professor, Politecnico di Milano

- ▶ The growth in sensor networks has generated the area of stream databases and processing. With semantic technologies, this will become “stream reasoning.” A sample application is

Urban Computing (pervasive computing in urban environments), where stream reasoning could address dynamic, real time problems in cities, e.g. changes in environment, accident notification, tracking movement of persons and goods.

Vojtech Svatek, Associate Professor, University of Economics Prague

- ▶ Semantics will be used in cost/benefit modeling – to help determine in which cases semantic technologies are worth using! For example, some are still sceptical in the database community about semantics in knowledge discovery, but we expect the benefits will soon outweigh the costs. There is a need for ‘metamorphic’ ontologies that can be automatically refactored according to need. Since ontologies will be the backbone of semantic applications, they need to bend along with application use!

14.4.2 Trends in Semantic Technologies by 2019

The experts found consensus on the gradual “mainstreaming” of semantic technologies as they achieve maturity in key areas such as automated ontology creation, web-scale reasoning, and personalization of information delivery. The new and increased demands in the Future Internet – the ubiquity of access, overload of information, and heterogeneous sources of data – would make semantics even more critical and spur their development and uptake. First application areas would start widespread usage of semantic technology in critical processes, most likely health care and automotive. Virtual worlds and sensor networks are seen as two IT trends which will not only become mainstream and widespread, but will find new uses for semantics. The experts, however, diverged on the exact timing and the market potential of each of these future applications areas.

14.4.2.1 Individual Visions of the Invited Experts

Philip Cimiano, Assistant Professor, Technical University of Delft

- ▶ Entitled “Semantics reaches the masses!,” the key trends were seen as data and application ubiquity, information overload and data redundancy. Semantics will improve search and information delivery, and by 2019 will have achieved measurable criteria in benchmarks, performance and usability. The “shallow web” (Web 2.0 data) will have merged with semantics to allow on-the-fly data processing from low level data (clicks, tags) and personalized/contextualized social information delivery. Metadata creation will be a fully implicit action from daily work and from context, e.g. sensors. In applications, particularly the automotive and healthcare sectors may have semantics in critical systems.

Fabio Ciravegna, Professor, University of Sheffield

- ▶ 3D and virtual Worlds can be in use for social participation, commercial simulation, complex experiments and cultural preservation. Semantics has the potentiality to be embedded into

objects in those worlds for interoperability, trading and interaction, and for trust and provenance.

David De Roure, Professor, University of Southampton

- ▶ Semantics will enable rapid application development over multiple legacy network information systems. For example, semantics make it possible for sensor networks to dynamically discover and integrate with one another and with other data sources, as well as for the rapid development of flexible decision support systems based on data from different networks. This will be based on production quality tools, network “mash-up” tools for non-experts and effective automation.

John Domingue, Professor, The Open University and President, STI International

- ▶ The growth in mobile and ubiquitous Internet devices will necessitate the development of a new network infrastructure for the Internet. Everything will become a service, and be tradable. Semantics are needed to enhance and enable planet scale networks, hence they will be an unavoidable part of the Future Internet.

Aldo Gangiemi, Senior Researcher, ISTC-CNR

- ▶ Semantics will come to underlie application development, and result in large scale CMS and “knowledge units” which will help online organizations to move from an “image” to an “identity,” and reducing the time for the delivery of personalized web-based applications. Ontology creation will be fully automated and high quality, leading to on-the-fly ubiquitous semantics, situation awareness and anomaly detection. Analogical representation and reasoning will make possible a semiotic web. Maybe semantics can solve some of the worlds problems: energy consumption, environmental sustainability, cultural integration...

Carole Goble, Professor, University of Manchester

- ▶ Semantics will be part of global, digital health systems, e.g. MyFamily – health information sharing between family members, Energy Balance Wristwatch – a personal device for sensing fat depositing and burning activities, Electronic Health Records – which can be shared under certain conditions to process health trends in a sample of people and make predictions, E-laboratories – where data from experiments and experts can be combined and processed to discover new information.

James Hendler, Professor, Rensselaer Polytechnic Institute

- ▶ Trends are the increased import of “social context,” decreased importance of the machine (life in the cloud), changing concept of trust and privacy, and increased data intensity. Semantics will make possible better profile-based matching, scalable back-end reasoning technology, next generation personal information devices and context-aware end user applications. Agents will be taking care of many information tasks for humans, and avatars will make it out of games (e.g. doctor, librarian).

Guus Schreiber, Professor, Vrije Universiteit Amsterdam

- ▶ There will be greater context awareness, personal profiles “in the cloud,” XXL datasets and lightweight reasoning. Virtual worlds will be commonplace and we will have knowledge democracy (e.g. semantic Wikipedia for niche areas, cooperative semantic annotations, medical information access).

Elena Simperl, Vice Director, Semantic Technology Institute Innsbruck at the University of Innsbruck

- ▶ The network is the people’ Human intelligence will be channeled and exploited to resolve semantic computing tasks and integrate human with computational intelligence. Semantics will have been simplified w.r.t. language and interface, and ontology lifecycles will be part of the personal information management. A semantic “Web 2.0” will create metadata and ontologies through incentive structures and integration into games, virtual worlds, or office tools.

14.4.3 Trends in Semantic Technologies by 2024

In soliciting visions for 2024, it was recognized that forecasting accurately what the world will be like in 15 years from now is a very difficult proposition. How many people would have anticipated today’s Internet back in 1994? However, gradual trends have been identified for 2014 and 2019 which can serve as a starting point. In particular, two specific technological directions were proposed as growing in significance in the next 15 years:

The Internet of Things raises new challenges in research that arise when (nearly) everything is on the Internet and consuming and providing data, and reflects the potential of semantic technologies to organize and mediate between the large scales of data which result.

Ubiquitous Data Streams follows from this to consider the challenges created by the data streams those things that the Internet of Things would be generating, and how the information in those streams may be extracted and shared with others in an automated fashion taking privacy and trust issues into account (supported by semantics).

14.4.4 Points Raised in an Open Discussion, Inspired from the Proposed Trends

- The gradual convergence of the digital and real world, and semantics as a means to model complex events
- The dangers of omnipresent data and the need for (semantic) technology to identify untrusted information, sensor errors, deception, etc.
- The need to fill the gap between knowledge experts and the “farmer in the field” whose knowledge should be modeled, if semantics are to ever come out of their high tower and down to, for example, agriculture in the developing world

The vision for 2024 which formed among the experts went beyond the expected results for 2014 and 2019 (maturing and subsequent use of semantic technology in mainstream personal, social, and enterprise activity), with a recognition of the breakthroughs which may be reached in terms of capturing semantic data about almost everything in the world and making them available everywhere and at every time. Combined with the potential for large-scale inference, it may be that too much knowledge can become accessible, and there is an imperative for semantic technology researchers to also consider and tackle the challenges of preserving privacy and trust in a semantic world (where previously unforeseen knowledge becomes inferable from combining heterogeneous and separate datasets). A more detailed insight into this longer-term vision is given in the Future Issues section.

14.5 Related Resources Including Key Papers

14.5.1 Semantic Roadmaps

This section contains some of the key publications of the past years that have studied the state of the art and progress in Semantic Web technologies and made predictions for the future of semantic technology on the Web and in enterprises. These publications in particular have informed the expert discussion around the future of semantics and are complementary to the visions and trends identified in this chapter, which updates and continues this discussion.

Finding and Exploiting Value in Semantic Technologies on the Web. David W Cearley, Whit Andrews, and Nicholas Gall. May 2007. Gartner Report G00148725.

- ▶ This Gartner report studied the Semantic Web and proposed that Semantic technologies' initial value will come from incremental improvements in finding documents in unruly content environments, which would take place over the following 10 years. By 2017, the report predicts that the Semantic Web will be fully evolved and the majority of Web sites on the Web will contain semantic markup.

The Semantic Web Vision: Where are We?. Jorge Cardoso, University of Madeira. September/October 2007. IEEE Intelligent Systems, vol. 22 no. 5, pp. 84–88.

- ▶ This journal report summarizes the results of one of the first surveys of experts on semantic technologies with a significant sample size. The survey asked 627 Semantic Web researchers and practitioners how they were using and interrelating semantic technology.

Knowledge Web Technology Roadmap. Roberta Cuel, Alexandre Delteil, Vincent Louis and Carlo Rizzi. December 2007. Published at: <http://kw.dia.fi.upm.es/O2I/menu/KWTR-whitepaper-43-final.pdf>

- ▶ The EU Network of Excellence Knowledge Web brought together key research organizations driving semantic technology R&D in Europe. As such, it was the ideal context for a roadmapping activity to extract insight into the current state of the art, the gaps, and hence the necessary research roadmap to bring semantic technology in the following years

to maturity, and achieve enterprise uptake. The Delphi and Focus Group methodologies were used with a select group of experts to generate a vision for technology maturity in both the research and industry contexts, and make a set of recommendations for future semantic technology R&D.

Semantic Wave 2008 Report: Industry Roadmap to Web 3.0 & Multibillion Dollar Market Opportunities.” Mills Davis. February 2008. Executive summary at: <http://www.calt.insead.edu/eis/WebTrends/2008semanticwebreport.pdf>

- ▶ The Semantic Wave report represents the most comprehensive industry study of the Semantic Web to date, intended for business people who want to understand better semantic technologies, the business and market opportunities they represent, and the ways this will change how we use and experience the Internet. It predicts multibillion dollar markets for semantic technology-based products and services, with consumer Internet being a major growth area, and enterprise adoption increasing dramatically.

What Semantic Technology Means To Application Development Professionals. Dave West, Forrester Research. October 2009.

- ▶ This report for Forrester reflects the emerging maturity of semantics. While Gartner in 2007 sees semantic technology uptake as a future vision, Forrester in 2009 opens with “semantic technology is now ready for increased but focused use. . . during the past 18 months, major changes have occurred in the application of semantic technology to real business problems.” It calls on application development professionals to build a pragmatic strategy for slowly integrating this new way of thinking into those projects where information provides the greatest value.

14.6 Predicting the Next 15 Years of Technology [31]

Looking from now to the next 15 years, the question being asked in this section is: What can be expected from technological developments in the next 15 years? What will be the major trends, the most significant breakthroughs? Making future predictions about technology is notoriously difficult – how much about today’s Internet-saturated world would have been predictable in 1994, at the beginning of the Web? It may be necessary to start believing seemingly impossible things. According to William Gibson – who coined the idea of cyberspace in his 1984 novel *Neuromancer* a decade before the Web – “the future is already here. . . it is just not evenly distributed” [32]. So arguably it is not necessary to anticipate some *deus ex machina*, or expect any magic silver bullets:

- It is unlikely that there will be any new technology not knowable today.
- It is unlikely that any new technology will have a major impact on things in the next 15 years.

Rather, the first step should be to study what will be the enabling technology trends. Many new technologies emerge to enable new applications when combined with existing

technology trends, for example, in combination with significant falls in technology adoption or purchase cost. For example, cheap hard disks (below \$20) made the iPod possible. So which trends can be relevant for the future of the Internet and computer technology?

- *The number of Internet users* will continue to climb – as of June 2008, there were an estimated 1.463 billion Web users, of a total world population of around 6.77 billion [33]. Particularly in developing populous regions of Asia, Latin America, and Africa, there is a strong potential for Internet usage explosion.
- *Moore's law* says that the number of transistors that can be placed inexpensively on an integrated circuit doubles approximately every 2 years, meaning that the limit of 100,000,000 transistors around the year 2000 has already become 1,600,000,000 transistors in 2008.
- *Disk space costs* mean that in 2003, 60 GB space cost less than \$100. Yet 18 years ago, this amount of space would have cost me \$6,000,000. The cost and scale of digital storage continues to drop: \$100 can now (in mid-2009) purchase me 500 GB of storage [34].
- *The size of the digital universe* means that, in 2007, the amount of information created, captured, or replicated exceeded for the first time the available storage. It is believed that by 2011, almost half the digital universe will not have a permanent home [35].
- *Internet bandwidth* is increasing by 50% per year, so that the high-end user's 10 million bits/s connection in 2007 will be over tenfold – more than 100 million bits/s – by 2013 [36].

What will become possible in light of these trends? With superfast broadband speeds, it is foreseeable that Internet-based HDTV with 10 Mbps broadband, 3D-TV at 40 Mbps broadband, and all sorts of multiple applications over the Internet pipe as bandwidth crosses 100 Mbps [37] will become a reality. Yet such visions are quite light, when one considers Ray Kurzweil's "Law of Accelerating Returns." This law states that technological change is exponential rather than linear, meaning that:

- ▶ We won't experience 100 years of progress in the 21st century; it will be more like 20,000 years of progress (at today's rate) [38]

Kurzweil predicts that the equivalent of 4,000 years of technological advancement will occur during the first 2 decades of the twenty-first century. Ray Kurzweil's prediction is, however, considered very controversial and contested by several researchers. See for example the Special Issue of IEEE Spectrum on Singularity published in June 2008 (<http://spectrum.ieee.org/static/singularity>). It is, however, reflective of the disruptive shift in human activity and society which has been brought about by the Internet and the expectation that one's future will be affected – in many ways – by the ubiquitous presence of this network and the data and services it carries:

- ▶ There is one only time in the history of each planet when its inhabitants first wire up its innumerable parts to make one Large Machine. Later that Machine may run faster, but there is only one time when it is born. You and I are alive at this moment. [39]

While how fast the “Machine” may run in the future will always be up for debate, its existence is not contestable and already plays a visible and invisible role in societies and economies every day. Futurists such as Kurzweil may think at the extremes of human imagination of the technological future, yet they allude to the extreme possibilities that might be realizable in a near-future where current barriers such as computing power or storage space have been removed. In Lewis Carroll’s “Alice in Wonderland,” the Queen tells Alice that she has believed as many as six impossible things before breakfast. There are many impossible things about the world in 2024 that may need to be believed before it is plausible to even begin to envision what role semantic technologies could play in this seemingly impossible future. For example, different sources have proposed:

1. “It is not hard to imagine a server computer storing information associated with every cubic meter of the earth’s surface” [40].
2. Every square meter of atmosphere hugging the earth may be filled with unseen nanodevices designed to provide seamless communication and surveillance [41].
3. Humans will have nanoimplants, facilitating interaction in an omnipresent network [41].
4. Everyone will have a unique Internet Protocol (IP) address [41].
5. AI entities will achieve a PhD, be awarded a Nobel Prize and given the right to vote [42].
6. War will be waged with more robots than human soldiers, and smart bacteria weapons will be used that alter enemy behavior [42].

While some of these predictions (or all) may seem unimaginable today, they cannot be described as unforeseeable – all can be seen as potential consequences of the trends in Internet and technology that are observable today. Every technology and application which succeeds can be considered to have passed the Technological Darwinism test: they will have been driven by the need to enhance an already existing basic human need, such as the need to communicate, exchange goods and services, shape own identities, and so on [43]. However, technology is arguably something that is morally neutral: it can be used equally for good as for evil, depending on its human “master.” So while the drivers for technological progress may be based on human benefit, the consequences may prove to be less beneficial to humankind.

This speaks strongly, for example, for prediction 3, where – like pen and paper today – future technology could allow for nanotechnology which becomes an extension of the human body. Humanity has always been very adept at dovetailing its mind and skills to the shape of the current tools and aids, and when technologies actively, automatically, and continually tailor themselves to humanity, just as humanity does to them, then the line between tool and user becomes flimsy indeed. Such future technologies will become less like tools and more like part of the mental apparatus of the person. They will remain tools only in the thin and ultimately paradoxical sense in which humanities’ own unconsciously operating neural structures are tools . . . “we are naturally born cyborgs” [44].

So, through studying the technology trends which are observable today, projecting their exponential future growth and imagining “impossible things” which could arise

as a natural consequence of that, a challenging vision for 2024 is arrived at, in which not only semantic technologies are expected to have achieved mainstream adoption, but also the world into which they are adopted has also changed greatly. This leads to a final, and significant, question: What role will semantics play in the world of 2024?

14.7 Roadmap for Semantic Technology to 2024

Core Semantic Web technologies such as ontology development and reasoning are seen as being mature as of 2009, meaning they are already leaving the research lab and building commercial applications which are mature enough for enterprise usage. Over the next 5 years, research will address some critical gaps for industrial uptake – scalability of reasoning, using lightweight ontologies, and maintaining and evolving ontologies – and semantic technology will be ready for the mainstream. Some longer-term aspects which can be linked to a need for more computing power combined with semantic technology and are medium-to-long-term challenges include reasoning at the scale of the Web, semantic data mining, for example, of real-time data streams, semantic context, and automated annotation. These will be necessary to first address data outside of controlled walled gardens (e.g., inside an enterprise, to handle specific aspects) and bring semantics effectively to large-scale data, whether they are data being generated in specific application domains or the Web. Hence, while semantics in the enterprise may be moving into mainstream markets over the next decade, the Semantic Web as it was originally thought of as an extension of the existing Web will first emerge (as a network underpinned by rich ontological reasoning) after this period (the next decade on the Web will be marked by the growth of structured data and APIs, in particular Linked Open Data and SPARQL as access mechanism, with some simple reasoning being added, e.g., sameAs).

Emerging technology trends will also incorporate aspects of semantic research which have been identified by the experts as relevant to the full realization of that technology some time after those aspects have reached maturity and begun to leave the research lab, being taken up by the early adopters. Considering each emerging technology in turn:

- *Security and privacy* will emerge in the next years driven by specifications such as OpenID and OAuth. Semantic enrichments of security and trust models are vital for future dynamic data and service usage situations, and will probably gain traction in commercial infrastructures in 5–10 years as research work is mature and the limitations of non-semantic approaches become clear. In particular, uptake will be driven by the new challenges arising from Web ubiquity (incl. Internet of Things), scale and heterogeneity of data streams (incl. from sensors and virtual worlds), and dependence on the social Web and collective intelligence (where trust and provenance play a vital role). Semantic security and trust will mature by 2019, and face issues of transfer into the already emerged technologies but it will be ready to play a fundamental role in the then emerging Internet of Services. A related requirement here is scalability, which will be less important for early semantic security solutions

(applied in the enterprise at smaller scale) but is vital for a Web-wide semantic solution to identity and privacy.

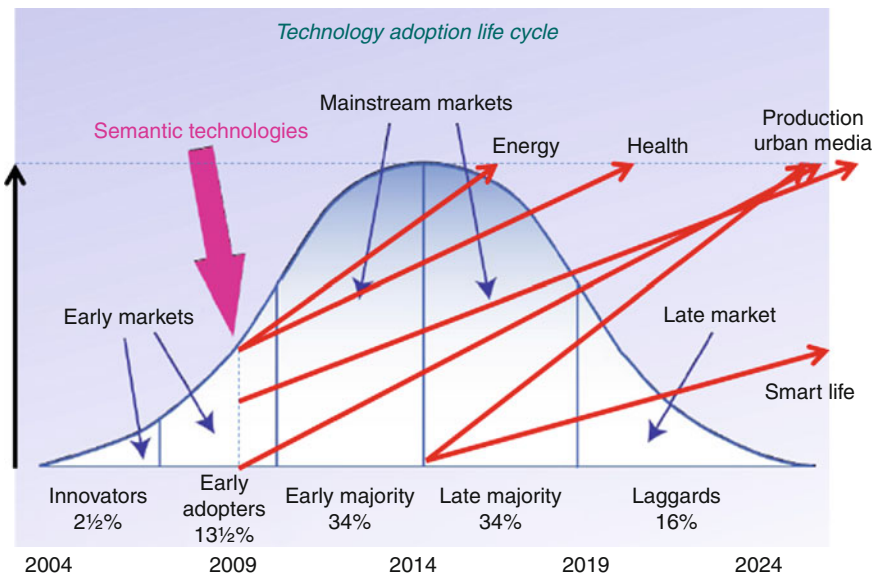
- *Sensor Networks* will become omnipresent in the next 5 years, for example, in urban computing one can expect the use of sensors in city infrastructure to regulate all aspects such as traffic flow, lighting, pollution, etc. Current work on semantic sensors can lead to uptake of semantic technology in the emerging technology provided various requirements can be demonstrably met by semantics: scalability, service execution environments, real-time reasoning, and object (sensor) description. Object description and service execution environments are being addressed well in the research while in the former, industry standardization is missing and in the latter, maturity for critical applications ([45], [46]). Scalability and real-time reasoning are still open challenges, where good progress is expected in the next 5 years; however, performance at web-scale will probably remain inadequate for 10–15 years yet.
- *Virtual Worlds* will emerge as the digital variant of the real-world intelligent sensor networks. Object description for 3D worlds as a research topic requires further focus. The other key requirement is large-scale knowledge management, to maintain a semantic view on both the state of the virtual world and of the real world. This will be achievable in the short term for large-scale, centralized approaches; world-scale and distributed solutions will remain a challenge for 10–15 years' research.
- *Internet of Things* is a natural consequence of Net connectivity across all devices. Large-scale knowledge management is a prerequisite for a semantic Internet of Things. However, the complexity of the knowledge may be less than in sensor networks and virtual worlds; hence, efficient lightweight ontologies and reasoning may be the key enabler. Context will also be needed to ensure rich adaptive services and applications, and there remains the main challenge of how to model context and develop services which can adapt to it, something predicted for the middle to long term of current research.
- *IP Television* reflects the convergence of audio-visual media and the Internet, and will be semantic as long as media descriptions are standardized – something which is ripe to begin, as research within W3C has been ongoing since 2001 and ontologies such as COMM exist as a basis (<http://comm.semanticweb.org>).
- *Social Technology* will evolve into Collective Intelligence to enable the decentralized performance of tasks through the involvement of Web users could make more use of semantic technology, something which has been largely ignored in Web 2.0 circles. Key here is the maturing of combined Social Semantic Web approaches to knowledge capture and reasoning. Challenges in annotation and ontology development are eased by the involvement of the community and semantic means to do this are emerging from research, while support for non-textual content may be 2–3 years behind. Lightweight ontologies and reasoning will mature also in the period before the emergence of Collective Intelligence, and should be a main driver of Collective Intelligence solutions.

A key bottleneck for the next 10–15 years will be the scalability of semantic technology in comparison to the scalability of the surrounding technology infrastructure: web-scale

or world-scale frameworks for sensor networks, Internet of Things, or Internet of Services are expected to emerge without semantics earlier than the timepoint where semantic technology may work as well at that scale.

In terms of application areas, the expert review has led to the following observations:

- *Energy* has key business drivers for the uptake of semantic technology and will increase uptake as the technology matures, particularly with respect to scalability.
- *Production* shares key business drivers such as data integration, but also faces greater challenges in terms of cross-enterprise communication. Besides the business barriers which occur here (e.g., agreement on common ontologies), it is dependent on semantic technologies which are developing later with respect to industry maturity: rules, trust, and semantic SOA. First when these technologies can be demonstrably used to solve real production sector issues, can one expect uptake to start at a measurable level.
- *Media* also has key business advantages from a transition to use of semantic technologies, yet very strong challenges are faced in terms of multimedia annotation. There will be gradual uptake pushed by emerging standards and mature tools, with a wide breakthrough first achieved when almost full automatized annotation becomes possible to handle the even larger scales of media being produced (not necessarily machine-automated, but also human-sourced collaborative annotation may become a solution here).
- *Health* is a fellow early adopter of semantic technology, especially in the research labs. In terms of wider uptake, for example, in end patient systems, it can be expected that



■ Fig. 14.3

Time for application areas mainstreaming semantic technology

the industry will be quite conservative in end implementation and will wait on the application of semantics in critical systems in other domains such as energy.

- *Smart Life* will be driven by the Internet of Things, while semantic solutions will be dependent on semantic technology expected to mature later, such as context models and the efficient processing of actionable knowledge. Full implementations based on semantic services will be long term. Also, the uptake of technology in new areas such as in-car, clothing, etc., can be expected to be slow in any case.
- *Urban Computing* will be driven by Sensor Networks, yet the technological challenges it faces are already well known and act as a driver for the use of semantic technology. Given that the application area will develop after the maturing of the semantic technology which can address its challenges, semantic uptake should be rapid, with scalability being the main bottleneck over the first decade.

This results in the following (▶ [Fig. 14.3](#)), mapping the state of semantic technology in enterprises in general (based on the W3C adoption life cycle in [3]) to a time frame (based on the observations of this chapter) and adding individual uptake growth by sector (with the vertical axis representing the level of uptake in the sector up to reaching the mainstream market).

Based on a continuing maturation of the semantic technologies and a conservative view of enterprise adoption, first mainstream breakthroughs of semantics in industry may be expected in the period 2016–2021, where research reaches fruition in early adopter sectors such as energy and health care.

By 2024, semantic technology could be as ubiquitous as the Web is today, and combined with the capabilities of Web scalability, real-time reasoning, and world-scale knowledge management, there are both exciting new possibilities ahead as well as brave new challenges.

14.8 Conclusion: A Brave New World

This closing chapter of the first volume has cast an eye over the future trends in semantic technologies; focusing on the next 5, 10, and 15 years (it should be clear to the reader that, at the foreseeable rate of change, even 15 years is a challenge to envision) the opinion of a wide range of experts in the semantic technologies community has been solicited and provided. Putting this in context of the general expected trends in technology and society, developments in semantics will move hand-in-hand with wider trends such as the growth in network capacity, computer performance, and disk space.

▶ [Figure 14.4](#) gives an example of this: as networks, computers, and disks grow, semantics will move from maturity in managing knowledge in large organizations, into city scale, and then capture more complex objects (3D), with the possible limit being capturing information about the whole known world. ▶ [Figure 14.5](#) shows how emerging mature semantic technologies contributes to this: semantic sensors represent an integration of sensor networks and semantic technologies to allow not only for the ability to sense

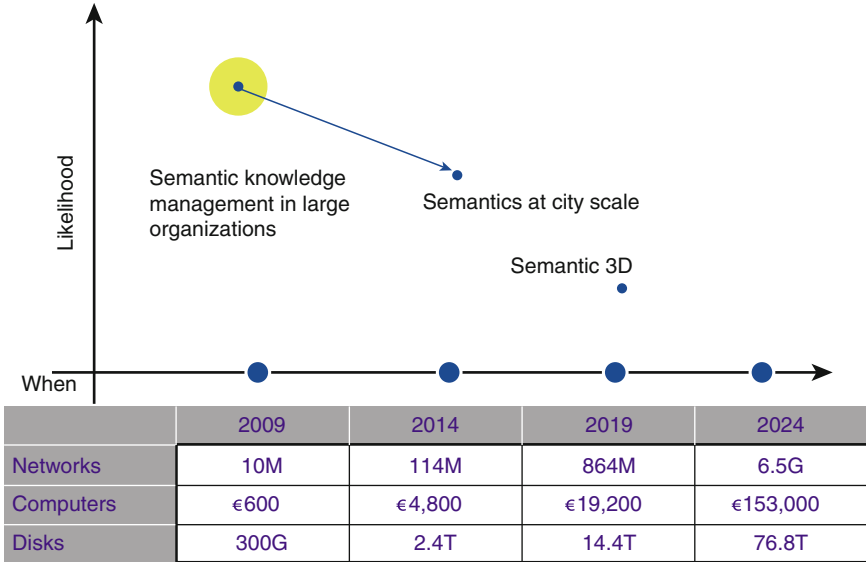


Figure 14.4 Likelihood of semantic technology developments

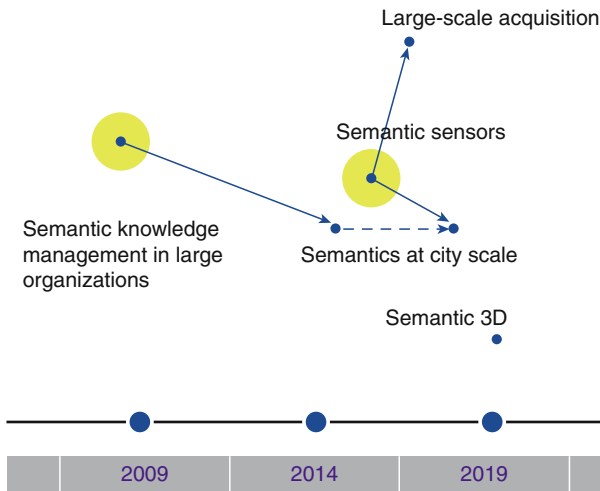


Figure 14.5 Enabling large-scale acquisition and semantics at city scale

the environment but also to intelligently make sense of it. Given their mainstream emergence in circa 5 years, according to the expert opinion, they will drive both the large-scale acquisition of semantic data and the semantics at city scale.

Emerging factors such as semantic sensors, effortless acquisition of data at large scale, the ability to handle huge amounts of data, and the exponential growth in bandwidth

(reaching 6.5 Gbps), computing power (10 billion devices online with typical specs of 1 TB of RAM and 768 GHz CPU), and storage space (76.8 TB) lead to the expectation that by 2024 one will be seeing collections of semantic data at world scale. To get there, and to derive social and economic benefit from it, there is still a wealth of research to be carried out to tackle challenges such as the scale and dynamicism of data, as well as provenance, trust, and security.

Combining these expert visions for semantic technologies, applications, technologies, and a broader vision for the world in 2024 has led to envisioning the next trends in semantic technologies, embedded in other social and technology trends around them, and come to a daring prophesy for a world (with semantics) by 2024, provided research challenges can be overcome.

14.9 Cross-References

- eBusiness
- Multimedia, Broadcasting, and eCulture
- eScience
- Social Semantic Web

Acknowledgments

This chapter includes summaries of the results of the STI International Roadmapping Service, in which a number of expert workshops took place over the period of September 2008 to July 2009 to gather experts' visions on the future trends in semantic technologies, and which would have not been possible without the following people:

The Roadmapping Service Co-ordinators Raphael Volz, Fabio Ciravegna and Rudi Studer

The presenters at the first workshop: Diego Berrueta, Michael Brodie, Fabio Ciravegna, Emanuelle Della Valle, Marko Grobelnik, Dunja Mladenic, Lyndon Nixon, Rudi Studer, Vojtech Svatek

The presenters at the second workshop: Philip Cimiano, Fabio Ciravegna, David De Roure, Stefan Decker, John Domingue, Aldo Gangiemi, Carole Goble, Mark Greaves, James Hendler, Ivan Herman, Guus Schreiber, Elena Simperl

The presenters at the third workshop: Andreas Abecker, Catherina Burghart, Fabio Ciravegna, Graham Hench, Elena Simperl, Rudi Studer, Ioan Toma

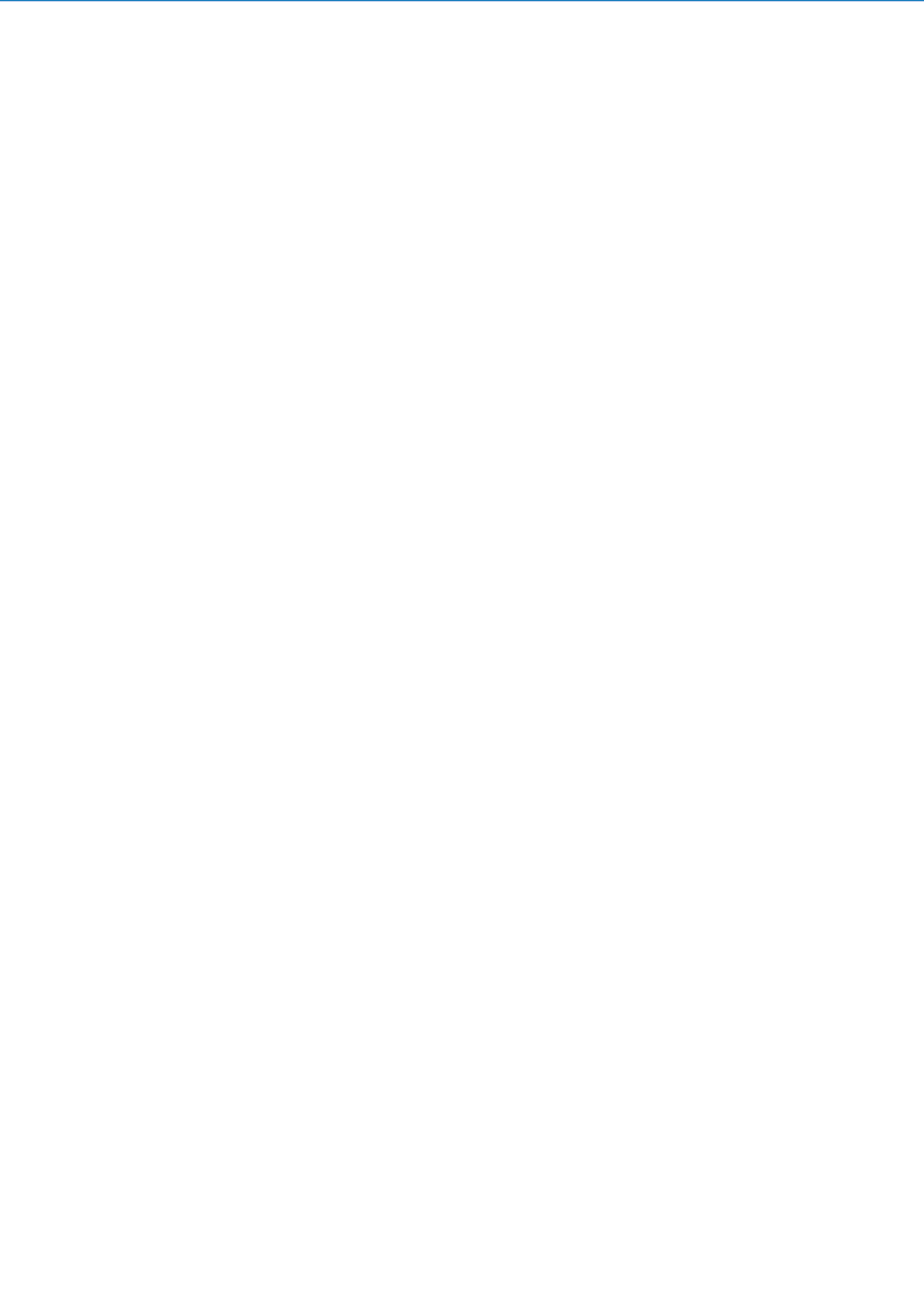
The Semantic Technology Institute (STI) is an association of leading industry and research organizations in semantic technologies. More details about STI can be found at <http://www.sti2.org>. The STI Roadmapping service is one of a number of member services set up and carried out by STI; for more see <http://roadmap.sti2.org>, which includes links to videos and presentations from all three roadmapping workshops.

References

- Computer Science and Telecommunications Board, National Research Council of Canada: Evolving the High Performance Computing and Communications Initiative to Support the Nation's Information Infrastructure. National Academy Press, Washington, DC (1995) (Non-Semantic technology data from http://www.nap.edu/openbook.php?record_id=10795&page=6 and 7)
- Gartner: An explanation of the Hype Cycle and its five phases. <http://www.gartner.com/pages/story.php.id.8795.s.8.jsp>
- Herman, I.: Semantic web adoption and applications. Presentation, W3C Recommendation. <http://www.w3.org/People/Ivan/CorePresentations/Applications/> (Oct 2009)
- Moore, G.: Crossing the Chasm. Harper Business, New York. <http://www.amazon.com/Crossing-Chasm-Marketing-High-Tech-Mainstream/dp/0066620023> (1999)
- ZDNet: Semantic web on verge of commercial viability. ZDNet News. <http://news.zdnet.co.uk/internet/0,1000000097,39460401,00.htm?r=1> (2008)
- Davis, M.: Semantic wave report: industry roadmap to web 3.0 and multibillion dollar market opportunities, Project 10x (2008)
- Chum, F.: Use case: ontology-driven information integration and delivery – a survey of semantic web technology in the oil and gas industry. <http://www.w3.org/2001/sw/sweo/public/UseCases/Chevron/> (Apr 2007)
- Abaas, R.: A SOA adaption strategy. Virtualization J. <http://virtualization.sys-con.com/node/565632> (2008)
- Aroyo, L., et al.: CHIP project @ Rijksmuseum Amsterdam. <http://www.chip-project.org/presentation/UvA.pdf>
- European Commission: Semantic interoperability, SemanticHEALTH report. http://ec.europa.eu/information_society/activities/health/docs/publications/2009/2009semantic-health-report.pdf (2009)
- Chen, L., et al.: Semantic smart homes: towards knowledge rich assisted living environments. In: Intelligent Patient Management, vol. 189, pp. 279–296. Springer, Berlin (2009)
- Aruvians Research: Analysing automotive telematics. <http://www.reportlinker.com/p0199283/Analyzing-Automotive-Telematics.html> (2010)
- Della Valle, E., et al.: Challenging the internet of the future with urban computing. The OneSpace Workshop at Future Internet Symposium (FIS 2008). <http://emanueledellavalle.org/blog/2008/09/28/challenging-the-internet-of-the-future-with-urban-computing/> (2008)
- Van Damme, C., Hepp, M., Siorpaes, K.: FolksOntology: an integrated approach for turning folksonomies into ontologies. In: Bridging the Gap between Semantic Web and Web 2.0 (SemNet 2007), pp. 57–70 (2007)
- Spivak, N.: My Metaweb graph – the future of the net. http://novaspivack.typepad.com/nova_spivacks_weblog/2004/04/new_version_of_.html (2004)
- Spagnuolo, M., Falcidieno, B.: 3D media and the semantic web. IEEE Intell. Syst. **24**(2), 90–96 (2009). <http://www2.computer.org/portal/web/csdl/doi/10.1109/MIS.2009.20>
- Pitidakis, M., Salamine, P., Thalmann, D., Catalano, C.: State of the art report on 3D content in archaeology and cultural heritage, FOCUS K3D deliverable 2.4.1. <http://195.251.17.14/downloads/project-deliverables/D2.4.1-final.pdf/view> (2009)
- Khan, A.N., Hussain, M.: Application of Semantic Web in e-business and telecommunication. In: Proceedings of the 2009 International Conference on Advanced Computer Control (ICACC 2009), Singapore (2009)
- Hauswirth, M., Decker, S.: Semantic reality – connecting the real and the virtual world. Position paper at Microsoft SemGrail Workshop, Redmond (2007)
- Sheth, A., Henson, C., Sahoo, S.S.: Semantic sensor web. IEEE J. Internet Comput. **12**(4), 78–83 (2008)
- Toma, I., Simperl, E., Hensch, G.: A joint roadmap for semantic technologies and the internet of things. In: Proceedings of the Third STI Roadmapping Workshop, co-located at European Semantic Web Conference (ESWC 2009), Heraklion (2009)
- Constantijn, V.O., et al. (Rand Europe): The future of the internet economy: a discussion

- paper on critical issues. Prepared for The Netherlands Ministry of Economic Affairs, Cambridge, UK (2008)
23. Simperl, E., Toma, I., et al.: Future internet roadmap, service web 3.0 deliverable 1.1. <http://www.serviceweb30.eu/cms/index.php/resources> under Deliverables
 24. Artz, D., Gil, Y.: A survey of trust in computer science and the semantic web. *J. Web Semant.* 5(2), 58–71 (2007)
 25. Kurzweil, R.: *The Singularity Is Near*. Viking, New York (2005)
 26. Uren, V., et al.: Semantic annotation for knowledge management: requirements and a survey of the state of the art. *J. Web Semant.* 4(1), 14–28 (2006)
 27. Grobelnik, M.: Semantic technologies. In: *New York Semantic Web Meetup* (2008)
 28. Simperl, E., et al.: Achieving maturity: the state of practice in ontology engineering in 2009. In: *Proceedings of the Confederated International Conferences, CoopIS, DoA, IS, and ODBASE 2009 on the move to Meaningful Internet Systems: Part II (OTM 2009)*, Vilamoura. *Lecture Notes in Computer Science*, vol. 5871, pp. 983–991. Springer, Berlin (2009)
 29. Courtesy Elena Simperl
 30. Kiryakov, A.: Measurable targets for scalable reasoning, LarkC deliverable D5.5.1 (2008)
 31. Ciravegna, F.: Six impossible things (invited speech). In: *Proceedings of the Third STI Roadmapping Workshop, Charting the Next Generation of Semantic Technology, during the Sixth European Semantic Web Conference (ESWC 2009)*, Heraklion (2009)
 32. Gibson, W.: The science in science fiction, talk of the nation, NPR. <http://www.npr.org/templates/story/story.php?storyId=1067220> (1999)
 33. Internet in numbers 2008. <http://royal.pingdom.com/2009/01/22/internet-2008-in-numbers/>
 34. Bradford DeLong, J.: Shifting into overdrive. *Wired Magazine* 11.05 (2003)
 35. Gantz, J.F.: *The diverse and exploding digital universe*. IDC (2008)
 36. Nielson, J.: Nielson's Law of Internet bandwidth: <http://www.useit.com/alertbox/980405.html> (1998)
 37. BBC News: Virgin eyes 150Mb broadband speed. <http://news.bbc.co.uk/2/hi/technology/7961135.stm> (2009)
 38. Kurzweil, R.: The law of accelerating returns. <http://webdiary.com.au/cms/?q=node/2635> (2001)
 39. Kelly, K.: We are the web. *Wired Magazine* 13.08. <http://www.wired.com/wired/archive/13.08/tech.html> (2005)
 40. Rheingold, H.: *Smart Mobs: The Next Social Revolution*. Basic Books, Cambridge (2003)
 41. US National Intelligence Council: Global trends 2025. http://www.dni.gov/nic/NIC_2025_project.html
 42. The BT technology timeline 2006–2051. <http://www.btplc.com/innovation/news/timeline/static/timeline.pdf>
 43. Khor, Z., Marsh, P.: Life online: the web in 2020. <http://www.sirc.org/publik/Web2020B.pdf> (2006)
 44. Clark, A.: *Natural Born Cyborgs*. Oxford University Press, Oxford (2004)
 45. Compton, M., Henson, C., Neuhaus, H., Lefort, L., Sheth, A.: A survey of the semantic specification of sensors. In: *Proceedings of the Second International Workshop on Semantic Sensor Networks (SSN 2009)*, Washington, DC (2009)
 46. Fensel, D., Kerrigan, M., Zoremba, M.: *Implementing Semantic Web Services – The SESA Framework*. Springer, Heidelberg (2008)
 47. Cearley, D.W., Andrews, W., Gall, N.: Finding and exploiting value in semantic technologies on the web. *Gartner report G00148725* (May 2007)
 48. Cardoso, J.: The semantic web vision: where are we? *IEEE Intell. Syst.* 22(5), 84–88 (2007)
 49. Cuel, R., Delteil, A., Louis, V., Rizzi, C.: Knowledge web technology roadmap. <http://kw.dia.fi.upm.es/O2I/menu/KWTR-whitepaper-43-final.pdf> (2007)

Semantic Web Applications



15 Semantic Technology Adoption: A Business Perspective

V. Richard Benjamins¹ · Mark Radoff² · Mike Davis² · Mark Greaves³ · Rose Lockwood⁴ · Jesús Contreras⁵

¹Telefonica R&D, Madrid, Spain

²Ovum, London, UK

³Vulcan Inc., Seattle, WA, USA

⁴Technology Industry Analyst, London, UK

⁵iSOCO, Madrid, Spain

15.1	<i>Introduction</i>	622
15.2	<i>Bringing High-Tech to Mainstream Market</i>	623
15.2.1	Key Factors for Making Business with New Technology	623
15.2.2	Key Processes for Making Business with New Technology	624
15.2.3	Key Buyer Roles for Making Business with New Technology	625
15.3	<i>Specific Aspects of Semantic Technology for Making Business</i>	626
15.3.1	Semantic Technology has a Classic Problem: It is Hard to Explain	627
15.3.2	Bridging Vendors with IT Managers: The STE Strategy Map	628
15.3.2.1	Corporate Dimensions: How to Describe a Company in Semantic Terms	628
15.3.2.2	How Semantic Technologies Support Business Activities	628
15.3.2.3	The Semantic Map, and Why It Has Curves	629
15.3.2.4	How-To: A Simple Way of Finding Opportunities for STEs	630
15.3.3	Predicting the Roadmap: Where Are Semantic Technologies Going?	631
15.4	<i>Semantic Offering in the Market</i>	632
15.4.1	STE Suppliers with Offering in the Market	632
15.5	<i>Adoption Horizons of Semantic Technology</i>	637
15.5.1	First Phase: Short-Term Adoption of Semantic Technology	639
15.5.2	Second Phase: Midterm Adoption of Semantic Technology	639
15.5.3	Third Phase: Long-Term Adoption of Semantic Technology	640

15.6	<i>Conclusions and Summary</i>	640
15.7	<i>Cross-References</i>	641
	<i>Annex 1: STE Suppliers Company Listing</i>	644
	Company Selection	644
	STE-Related Web Pages	645
	EU-Funded Project and Commercial Reports	645
	Conference Assistants	645
	Company Description Fields	645
	<i>Annex 2: Company Listings by Sector, Technology, and Application Area</i>	655
	STE Companies by Sector	655
	STE Companies by High-Level Technology	655
	STE Companies by Area	655

Abstract: In the past decade, significant budgets have been invested in the development of Semantic Technologies. In this chapter, relevant factors are laid out for semantic technology adoption and a framework is provided for describing and understanding the value proposition of semantic technology from a business point of view. An overview will be shown of the current semantic offering in the market place, and this will be interpreted in terms of the framework. Finally, three adoption horizons will be introduced for when semantic technology can be expected to reach the mainstream market and what role it will play in organizations. Overall, the conclusion is that there is some uptake of the technology, but that the mainstream market is still not reached. This is partly because of a lack of understanding by industries of in what situations semantic technologies can add value to their business. This is the reason for introducing a new framework that is aimed at enabling industries to more easily relate semantic technologies to their business.

15.1 Introduction

The value proposition of semantic technology is to enable applications and the Web to expose more intelligent behavior. As the name says, semantic technology adds meaning to content in such a way that it becomes machine understandable. This makes it possible to delegate tasks to computers that previously needed the intervention of humans. The classic paper of Berners Lee, Hendler, Lassila published in *Scientific American* in 2001 [1] gives a clear example of this. In that paper, personal software assistants act on behalf of people to schedule an appointment, a process which requires contacting various other software agents, repeated interactions, and careful decision making. Since the late 1990s, much research effort has been dedicated to developing and maturing Semantic Web technology. This chapter analyzes, from the context of mid-2009, whether this sustained R&D effort has led to significant market acceptance. To what extent has the growing Semantic Web had commercial impact, what are the drivers for future adoption, and what can the semantic technology community do to increase uptake of this technology in the market? The recent announcement of Google's plans to use RDFa in their Rich Snippets product, Yahoo!'s launch last year of Searchmonkey, and Microsoft's acquisition of PowerSet are certainly indications of the steady mainstream market adoption of semantic technology.

Rather than speaking of Semantic Web, the term Semantic Technology will be used, since the application of the technology goes beyond its Web instantiations. However, a restriction is made to technologies which make essential use of the Semantic Web languages RDF and OWL, and their variants.

At a high level, there are three main application areas for semantic technology: Information and Knowledge Management, Enterprise Application Integration, and Social Semantic Web. In each area, semantic technology holds promise:

- The promise of semantic technology for information and knowledge management lies in helping people address a constant problem in today's information society, which is to have through sift to increasingly larger amounts of information, both in professional and personal lives. More and more time is spent analyzing increasing amounts

of available information in order to find the information that is needed. It is common for professionals to spend several hours a day on reading and replying to e-mail, searching and navigating the Web, and gathering relevant information to make the best decisions. There are also corresponding information-intensive examples in personal affairs – for example, booking a holiday often requires a substantial effort of information gathering on the Web and analyzing the corresponding information against personal constraints to find the best combination.

- The promise of semantic technology for application integration problems is to significantly reduce the effort associated with managing data interchange between applications. Semantic technology can be used to annotate application inputs and outputs and characterize functionality in a machine-understandable language. This opens the way to automatic service discovery and composition (in SOA-style architectures) and thus promises to significantly reduce the cost associated with system evolution and maintenance. When paired with a planning system, Semantic Web technology can deliver automatic composition and choreography of semantically annotated Web Services.
- The promise of semantic technology for Social Semantic Web applications is to allow people to have a better online experience. Thus, semantic technology is used to enhance the common human activities of content creation, publishing, linking data to other data, forming communities, purchasing satisfying things, browsing, socializing, dating, etc. One common thread in this application area is to enhance the effectiveness of advertising for products that are likely to be desired, either by automatically creating profiles or providing a better framework for people to create profiles of themselves.

The chapter is organized as follows. The first section reviews factors and processes relevant for bringing new technology to the market. Then, in the next section, a framework (semantic map) will be presented that allows businesses to understand the value proposition of semantic technology along with a characterization of business situations in which the technology is applicable. The next section surveys the current semantic technology offering as provided by 100 companies (2009), and interprets this offering in terms of the framework presented. Finally, the chapter ends with a description of three horizons according to which semantic technology is expected to enter the mainstream market. There are two appendices: a list of the 100 suppliers of semantic technology and a categorization of their offerings in terms of sector, technology, and area of application.

15.2 Bringing High-Tech to Mainstream Market

15.2.1 Key Factors for Making Business with New Technology

Having stated the promise of semantic technology, what are the factors to be taken into account when one wants to use it in business? Creating business value using semantic

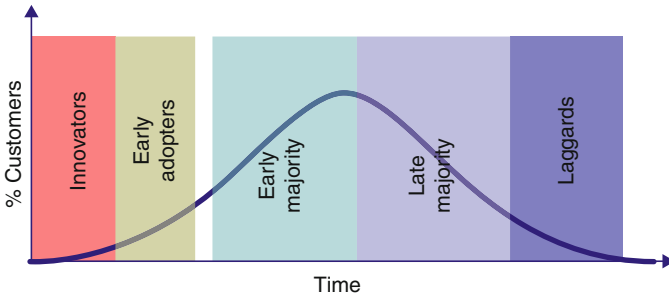
technology is in many ways no different from creating business value with any type of new technology. There are three important aspects to take into account:

- *Customers*: think about the customers that will use the new product or service that is enabled by the new technology. Technology as such does not provide anything: it is its capability to solve real customer problems that decides its business potential. Notions such as cocreation (that involves final customers throughout the product ideation, conception, development, and testing stages) and agile software development (in which software is developed in short cycles of a few weeks – sprints – where each sprint is validated with customers) are useful methodologies to become more customer centered.
- *Business models*: think about the business model of the new product and services. What is its place in the value chain, and what part of the value chain can it monetize? And of course, the costs of creating, selling, delivering, and maintaining the product needs to be less than the revenue it generates.
- *Technology*: how easy or difficult is it to have others replicate your product or service? Does the technology provide a competitive advantage compared to other technologies around? Is the IPR adequately protected?

The sweet spot for making business with new technology is to find the right balance between those three factors. Look at it as the intersection, that is, a product needs to be successful on all three aspects. A product that customers like and has a sound business model, but that lacks differentiating technology, is very easy to replicate by competitors, and if successful, will certainly take place (this is often referred to as *marketing* innovations). A product based on a differentiating technology and that customers love, but does not satisfy the business dimension, cannot be profitable simply because the costs will outweigh the potential revenues. And products that have sound business models and differential technology, but that customers do not like, will never reach a large market to become sustainably profitable.

15.2.2 Key Processes for Making Business with New Technology

But once there is an adequate product positioning with a potential market, the process starts to convince potential clients to adopt the product. Geoffrey Moore in his book *Crossing the Chasm* [2] gives an excellent account of the process of how markets adopt new technology. He describes the so-called technology adoption life cycle according to which high-tech products are adopted through subsequent stages from early markets to mainstream markets (➔ *Fig. 15.1*). Each of those markets has different types of customers. Early markets consist of innovators, real technical people who love technology for the sake of technology (the typical alpha and beta testers), and of visionaries, people who believe new technology can make a breakthrough in their business, and therefore are willing to take a certain risk. Mainstream markets are inhabited by two types of customers: pragmatists,



■ Fig. 15.1

The technology adoption life cycle of crossing the chasm [2] (explains how technology is adopted by the market place)

who believe that new technology can make a percentage difference in their business, but only are willing to adopt the technology if there is sufficient evidence from their peers that the product works and does not introduce additional risks. The conservative customers of the mainstream market only buy new technology products if not adopting them will cause them problems in their business. Finally, the late market consists of *laggards*, who never buy technology unless it is embedded in devices they already know. Moore claims that there is a gap between the early adopters and the pragmatists in the sense that many high-tech products reach early adopters but never make the big step to the mainstream market, and thus fail to be profitable. In his book he claims the percentage of failure to “cross the chasm” to be as high as 90%.

Another relevant notion to help making business with new technology is Open Innovation [3], which is the opposite of the NIH (not invented here) syndrome. Over the past decades, research has been globalized, as has happened to many business activities. Whereas in the 1970s it was possible to have the best researchers in a certain area all together at the same place (e.g., Xerox PARC, Bell Labs), in today’s globalized society that is virtually impossible. In order to still be able to tap on the best talent worldwide, successful research often requires depending on others for parts of the technology or business required for a new product. What better evidence for this than the acquisition activity of companies famous for their innovation pace such as Google, IBM, Apple, etc? In many of the research and development initiatives related to semantic technology, open innovation is not the main paradigm.

15.2.3 Key Buyer Roles for Making Business with New Technology

In order to bring any new technology product to the market (including products using Semantic Technology), a sine qua non condition is to convince clients. In many cases, clients will be enterprises and it turns out to be a complex process to sell to them even if

they are early adopters. Though details of sales processes are not relevant here, it is important that researchers and developers of Semantic Technology are aware of the following principal roles that are involved in selling new technology products to enterprises. Depending on the size of the enterprise, roles can be combined into one person or be distributed over several persons.

- *The users:* Users are the people who will enjoy the functionalities of the product. They should love the offered functionality, user interface (ease of use), speed, etc.
- *The IT department:* You convince a user with functionality, but the IT department decides on how to implement that functionality: with your product or with another (your competitors). Technical specifications of the product are relevant, but the enterprise's IT policy is equally important. It is not always the most popular (users) and technically superior product that is the winner.
- *The budget keeper:* Once users want the product and the IT department accepts it from a technological point of view, you still need a final “go”– decision that a certain amount of the company's budget be spent on the new functionality. Usually, in the case of new products, there is no corresponding “new” budget which you can draw from. Your product will compete with other products that may have nothing to do with your product, but simply may have higher priority.

In order to make a sale, all of those client roles have to be convinced.

15.3 Specific Aspects of Semantic Technology for Making Business

So far we have seen a variety of reasons why it is difficult for high-tech to reach mainstream markets. But there are also specific aspects of Semantic Technologies that may explain why it is hard for these technologies to be adopted by enterprises in the mainstream market. In a recent research project [4], a study was made to find this out by consulting both IT managers and suppliers of Semantic Technology. The results show that both vendors and executives have problems describing in simple, strategic terms how Semantic Technologies might fit within a business. This inability is slowing the uptake of STEs (Semantic Technologies for Enterprises) in companies – leaving the less alert IT manager behind operationally, and minimizing the market opportunity for vendor suppliers. In order to solve this barrier of STE uptake, the “STE Strategy Map” has been developed to help managers and vendors remedy the situation.

15.3.1 Semantic Technology has a Classic Problem: It is Hard to Explain

Operational executives want to know whether technology is suitable for their company and can make operations more efficient. Technology vendors want to reach the

operational executives and make a sale. The problem is, in the context of STEs, these two parties do not seem to have much common ground or language. Among many people interviewed, the word “Semantic” conjures one of two images: security software (among those who confuse it with “Symantec”) and the aging buzz-phrase “Semantic Web.” Many IT managers and executives simply do not understand the potential of semantics within their business. This would require a sit-down meeting with them in order to explain it. And, unfortunately, most vendors do not have such an opportunity.

The results are twofold:

- IT managers are missing that “a-ha!” moment where they see how STEs could provide them real benefits and have a place within their companies.
- With time passing, vendors for semantic technologies are watching as much of their technology offering risks becoming a commoditized add-on for major software vendors such as Microsoft and SAP.

Apparently, suppliers of STEs have not been successful in explaining for what businesses STE is applicable to provide benefits.

What is required then is a framework for discussion which will bring these two parties closer together in a much shorter period of time. This framework will need to explain:

- What kinds of companies need STEs, and what kinds do not
- What kinds of activities STEs can perform
- The point at which regular software ends and STEs begin
- The point at which STEs end and labor-intensive work begins

So far, such a framework has not been provided earlier and most researchers and suppliers characterize semantic technologies in terms of one or more fundamental components. These include standards such as Resource Description Framework (RDF) and Web Ontology Language (OWL), and technologies such as taxonomies and ontologies, and composite technologies such as natural language processing engines. The added value of the framework is that it takes a use-case scenario perspective: STEs help bridge business contexts or help employees execute more conceptual activities in the business often reserved for people. The framework is presented in the next sections and referred to as the “STE Strategy Map.”

15.3.2 Bridging Vendors with IT Managers: The STE Strategy Map

15.3.2.1 Corporate Dimensions: How to Describe a Company in Semantic Terms

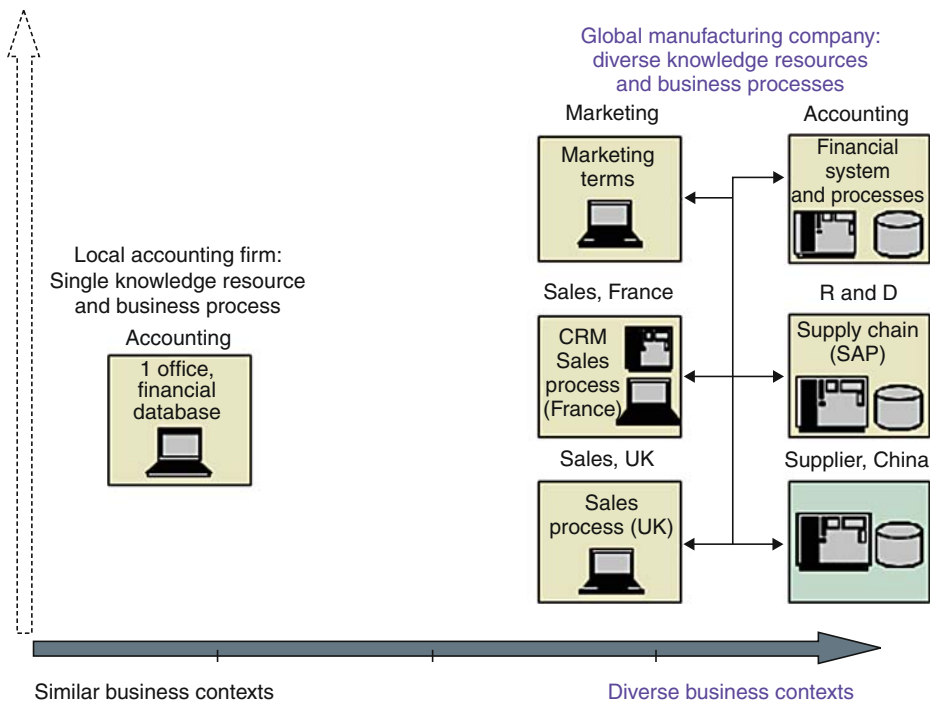
In order to first establish a context by which both IT managers and semantic vendors can describe a company, two basic dimensions are suggested: that of “context” and “concept.”

- *Defining “Context”*: Within an organization, a “context” has its own knowledge resources, business processes, and definitions for terms. A small accounting company may have some small functions for marketing and HR, but its primary context will be “accounting.” For a large multinational on the other hand, there are independent knowledge bases and processes for different departments (➤ [Fig. 15.2](#)).
- *Defining “Concept”*: Within an organization, certain activities may be conceptual, and others are not. Simple calculations are not conceptual, but developing a corporate strategy is (➤ [Fig. 15.3](#)).

15.3.2.2 How Semantic Technologies Support Business Activities

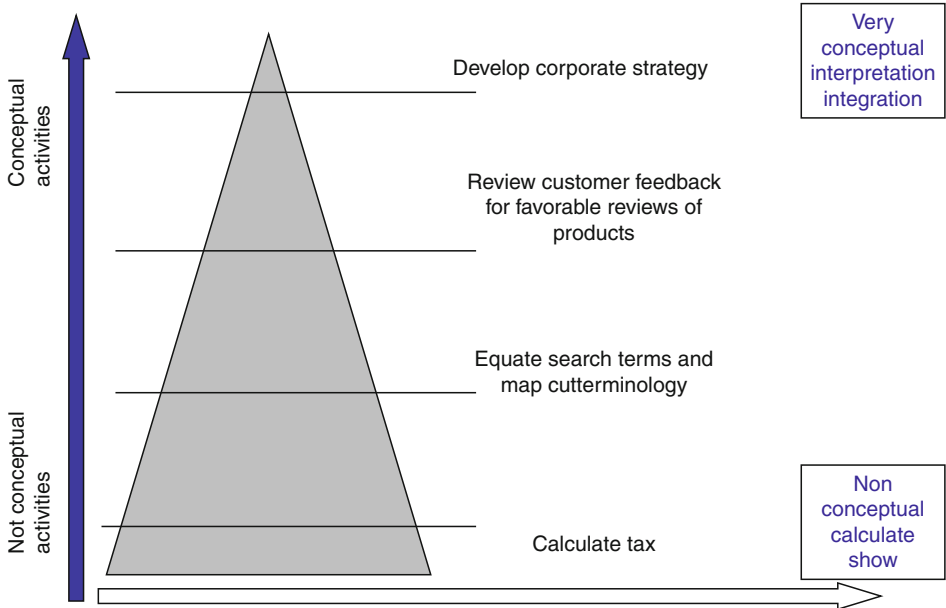
Businesses where STEs are needed and likely to flourish generally try to apply IT in support of one or both of the following types of activities:

- *Activities where concepts and conceptual understanding are important*: For example, reviewing legal documentation or contracts, or searching information for concepts



■ **Fig. 15.2**

Contexts within a business



■ Fig. 15.3

Conceptual and nonconceptual activities in a business

and data related to the topic of focus. Generically speaking, activities where the actor is trying to interpret, infer, augment, and aggregate information.

- *Activities that span a diverse range of business contexts:* For example, working across R&D, marketing, and operations departments to develop a new product line. Generically speaking, activities that help discover, harvest, create, present, transmit, or act across diverse knowledge resources and business processes.

Working with these basic dimensions, one can derive a few rules of guidance regarding the identification of which businesses are fertile grounds for STEs:

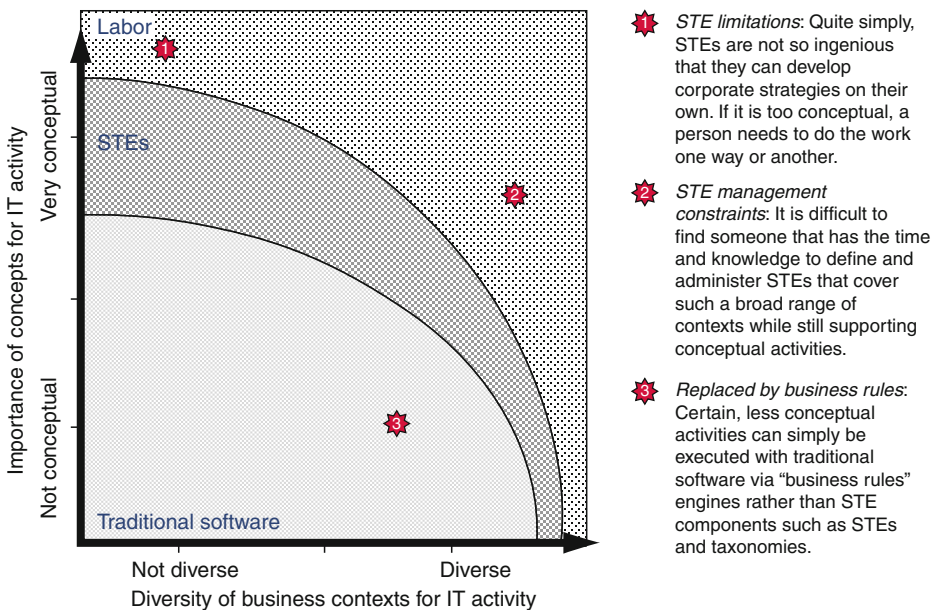
1. Semantic Technologies for Enterprises (STEs) are IT systems that support business activities that are some combination of “conceptual” and “contextual.”
2. Businesses that engage in neither conceptual nor diverse contextual activities are not fertile grounds for STEs.
3. Businesses that have IT systems, or need for IT systems, to conduct conceptual or diverse contextual activities, may be fertile grounds for STEs.
4. In order for STEs to flourish, the company must have a certain degree of maturity in its approach to information. It must have not only at least a moderately large volume of information, but view this information as an asset around which to build activities and processes. Otherwise, conceptual activities or multi-contextual activities cannot be run using IT.

15.3.2.3 The Semantic Map, and Why It Has Curves

Conceptually, while STEs need some element of Concept and/or Context, technological and operational constraints are also relevant for STE. Primary research and analysis has helped derive some basic constraints that put curves in the Strategy map (► Fig. 15.4).

Based on this, there are some fundamental tenants that CIOs, IT managers, and the vendors that sell to them, should remember:

1. *STEs have limits to their “intelligence.”* There are activities that are too conceptual for STEs. By the time the software has been developed to execute a highly conceptual task, the organization has burned up just as much if not more labor time in development and configuration as it would have in actually having a person execute the task. Furthermore, software is limited in how “artificially intelligent” it is. So there is an upper limit to how conceptual the task can be before a person must execute the task, making it a labor-intensive activity.
2. *Trying to get STEs to span to diverse contexts has operational and conceptual boundaries.* Conceptual tasks across multiple contexts become similarly difficult to implement using STEs. For instance, when working specifically with “ontologies,” there is potential for logical conflicts to arise within complex ontologies. Furthermore, few people in an enterprise have the breadth and depth of understanding necessary to create or administer ontologies that span multiple contexts. If they do, they seldom have the time to spare endlessly tweaking ontologies.



► Fig. 15.4

The curved map: constraints on semantic technologies

3. *If it is a simple enough activity, regular software will suffice.* If an activity becomes simple enough, business rules and “traditional software” can replace semantic technologies.

15.3.2.4 How-To: A Simple Way of Finding Opportunities for STEs

Once the corporate context and the Semantic Map are understood, they can be used to determine whether there are specific opportunities for STEs. To do this, it is possible, for instance, to consider various pain points that a company might have, as well as the likely IT responses to those pain points. It is then possible to determine whether the IT response falls in the strategic “sweet spot” for STEs:

Both the responses in [Fig. 15.5](#) to IT pain points (1 and 2) fall within the Semantic “sweet spot,” for which STEs can be effectively deployed.

15.3.3 Predicting the Roadmap: Where Are Semantic Technologies Going?

IT managers will undoubtedly wonder whether this technology is real and where it might be going in the long term. With a Strategy Map, they can at least begin the dialog with

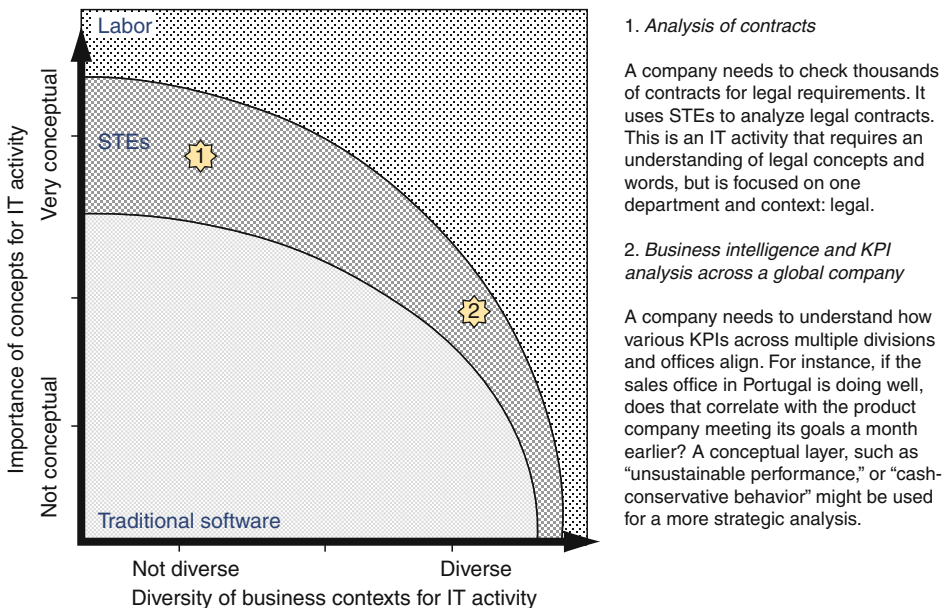


Fig. 15.5

Corporate pain points with semantic answer

vendors to become convinced themselves of whether STEs hold potential for their company. In general, research indicates that the field is real, that there exist certain pain points well suited for STEs, and that business is starting to take a look. As one of the interviewees pointed out, Semantic technologies are finding their way into the implementation of mainstream software:

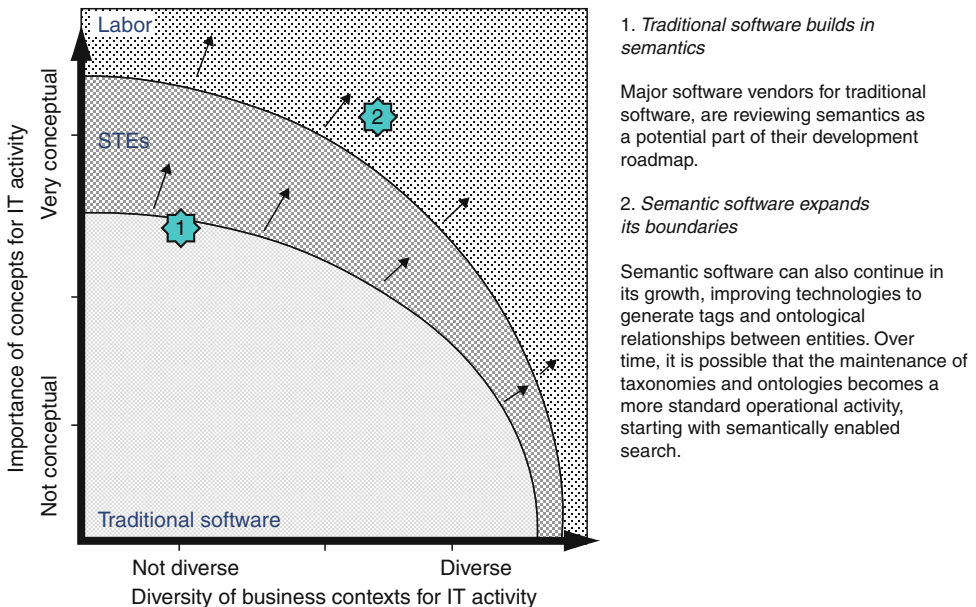
- ▶ “When we implement a new version of SAP, and our customer says ‘this is what we want,’ there are already RDF markups on the inside . . . in the new SAP implementations, RDF and semantic technologies are appearing as requirements.”

Depending on dialog and future business development efforts, Semantic technology may continue growing and winning a place on the market, or be swallowed up by the mainstream software vendors (◀ Fig. 15.6).

15.4 Semantic Offering in the Market

15.4.1 STE Suppliers with Offering in the Market

The previous section has identified the “area” where it makes sense to apply semantic technology; for solving problems that (1) are currently out of reach of traditional software (conceptual dimension) and (2) have a multi-context nature (diversity of business



■ Fig. 15.6

A roadmap for semantic technology

contexts for IT activity). This section surveys current semantic technology providers analyzing whether and how their offering fits the strategic map. In other words, the theory (semantic map) will be matched against the reality (the offerings).

About 100 companies are analyzed. The tables below group the providers based on geography: USA, Europe, and rest of the world. The appendix provides more detail on the companies in terms of sectors, technologies, and areas of application.

US-Based Companies

AdaptiveBlue	Cougaar Software, Inc.	Intellisophic	Semantech
Agilense	CyCorp	Invention-machine	Semantic Arts
Amblit	Digital Reasoning	Kirix	Semantic Research
Ask	DowJones	Knowledge Based Systems, Inc.	Siderean Software Inc.
AskMeNow	EMC Corporation	Knowledge Foundations	TeraDact
Autonomy	Endeca	LinkSpace	Teradata
BBN Technologies	Expressor	LucidMedia	Textwise
Boeing	Fortent	Metatomix	Thetus
Cambridge Semantics	Franz	Microsoft	Thomson Reuters
CheckML	Full Capture Solutions	Motorola	TopQuadrant
Cognition	Hakia	Oracle	WAND
Collexis	Hewlett-Packard Company	Progress Software	XSB
Connotate	IBM	Radar Networks	Yahoo!
Content Analyst	Image Matters	RiverGlass Inc.	Zepheira
Contivo	Intelius	Sandpiper Software	Zoominfo
Convera	Intellidimension	SchemaLogic	

EU-Based Companies

Aduna	Exalead	Nokia	SmartLogic
Altova	Expert System	Norkom	Talis
Aspasia Systems	Intelligent Software Components	Ontoprise	Thales
Biowisdom	iSense	OntoText	TrueKnowledge
Digital Pebble	Lingway	Reuse Company	Whatever
Empolis	Mandriva	SAP	Ximatrix
	Mondeca	Siemens	Zemanta

Rest of the World Companies

Netbreeze	Celtx	EffectiveSoft	Saltlux
IQSer	Reinvent	JustSystems	InfoSys
Ontos	Semantic System	Ontopia	

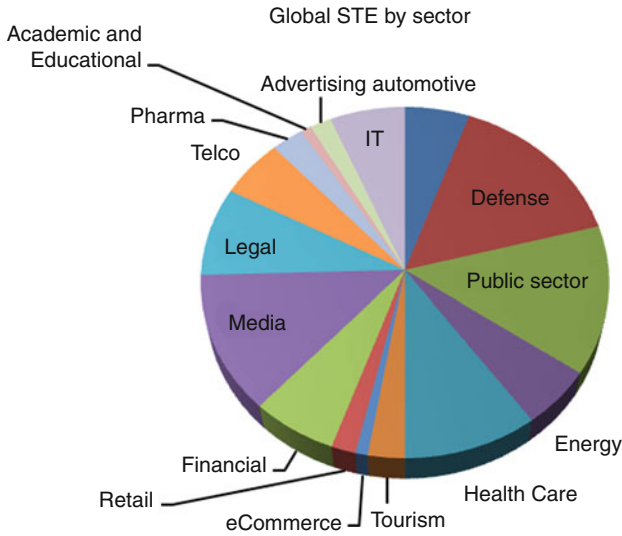
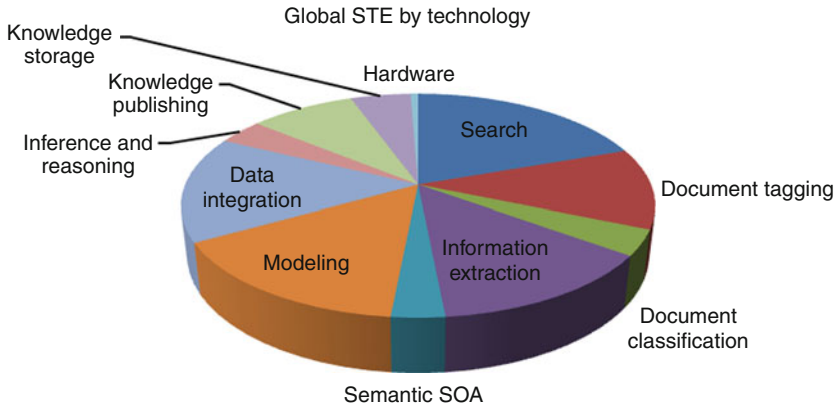


Fig. 15.7
Semantic offering according to sector

A first impression of what the current market offers in terms of semantic technology can be obtained by looking at the sectors the offerings apply to. The pie chart shown above (➤ [Fig. 15.7](#)) illustrates that the most represented sectors include Defense (15%), Public sector (14%), Media (13%), and Health care (10%). The fact that defense is number 1 is not surprising because traditionally, many new transformational technologies are first investigated and applied to defense, especially in the USA through the DARPA program. In terms of the Semantic Map, in the defense sector both the conceptual dimension as well as the “diversity of business context” dimension are relevant. Think, for example, of “Intelligence” applications that require conceptual capacity (beyond keywords) and a quick integration of different data sources. In the Media sector, the important problems are related to managing the content explosion, both textual and multimedia (photo, video, audio). Providing automatic support to managing this content requires conceptual capabilities of the solutions. In the public sector, problems are diverse and their automation involves both conceptual capabilities as well as putting together multiple business contexts, such as the integration or coupling of diverse IT systems. In the health-care sector, typical problems require the automatic integration (patient) data from different sources and formats, which is a manifestation of multiple business contexts.

If the offerings are looked at from a high-level technology perspective, the top five technologies (see figure below) covered include Search (20%), Modeling (15%), Data Integration (15%), Information Extraction (13%), and Document Tagging (12%). In the semantic map framework, four of those technologies are mostly related to the “conceptual” dimension, while Data Integration is related to the “diverse business contexts” dimension (➤ [Fig. 15.8](#)).



■ Fig. 15.8

Semantic offering according to technology

Probably the most relevant dimension for relating the current offering to the Semantic Map is the application area. An analysis of the offering reveals the following areas.

- *Knowledge Management*: Semantic technology allows the transformation of traditional knowledge management from a lexical to a conceptual level. More on Knowledge Management can be found in [Knowledge Management in Large Organizations](#).
- *Information Access*: Search engines supported by semantic technology improve the user experience by better understanding queries taking into account the meaning of terms. More on search engines can be found in [Semantic Web Search Engines](#).
- *Content Management*: Semantically extended content management systems may improve the user experience and the administration labor on web portals. The Media sector is one of the most promising sectors for STE adoption. A use case from the BBC World Cup 2010 website can be found in [Storing the Semantic Web: Repositories](#).
- *Document Management*: Similar to content management, document management may also improve using semantic technology for automatic document classification, retrieval, or information extraction. More on automatic information extraction can be found in [Semantic Annotations and Retrieval: Manual, Semiautomatic, and Automatic Generation](#).
- *Customer Relation Management*: Semantic technology for understanding user/customer's need, comments, claims etc.
- *Productivity Tools*: E-mail, desktop, or agenda management tools are experimenting with semantic technology for moving to a conceptual level.
- *Security and Intelligence*: Next-generation security and surveillance systems are based far more on the information content and meaning, rather than on signal processing and telecommunication defense.
- *Social Networks*: The formalization of social relations using semantic technology allows for a better automatic processing for preferences, recommendation, and advertisement.

- *Enterprise Application Integration:* Application integration is one of the biggest pain points of IT managers from the budget point of view. Integration using semantic data or Semantic Web Services may drastically decrease the needed effort for full IT infrastructure exploitation. More on Semantic Web Services can be found in [▶ Semantic Web Services](#).
- *Business Intelligence:* Related to the CRM area. Semantic technology may help in the intelligent integration of heterogeneous business knowledge coming from different departments for a consolidated business view on the company's results.
- *Business Process Management:* Similar to application integration, the formalization of enterprise business processes may allow for a cheap and quick adaptation of existing processes and the development of new processes according to business needs.

The figure below ([▶ Fig. 15.9](#)) shows how these application areas can be plotted on the Semantic Map. From top to bottom, until “productivity tools” the areas are mostly conceptual. From “business intelligence” to “Enterprise Application Integration,” the areas have a more “diverse context” character.

If the offering is analyzed from the application area perspective, as can be seen in the figure below ([▶ Fig. 15.10](#)), the top five are: Information Access (32%), Knowledge Management (19%), and then Enterprise Application Integration, Content Management, and Security and Intelligence, all three with 11%. This is consistent with the “technology”

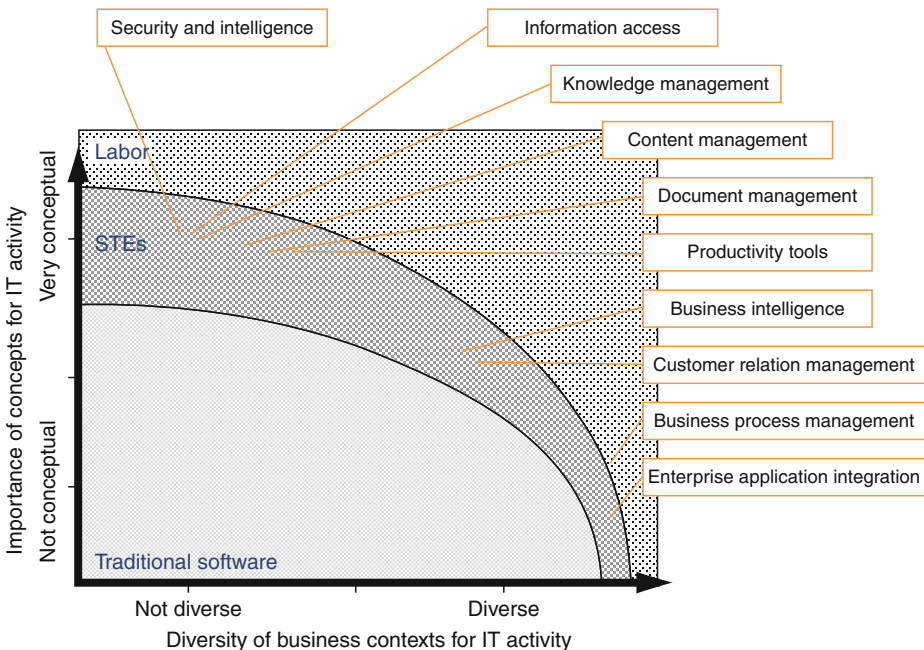
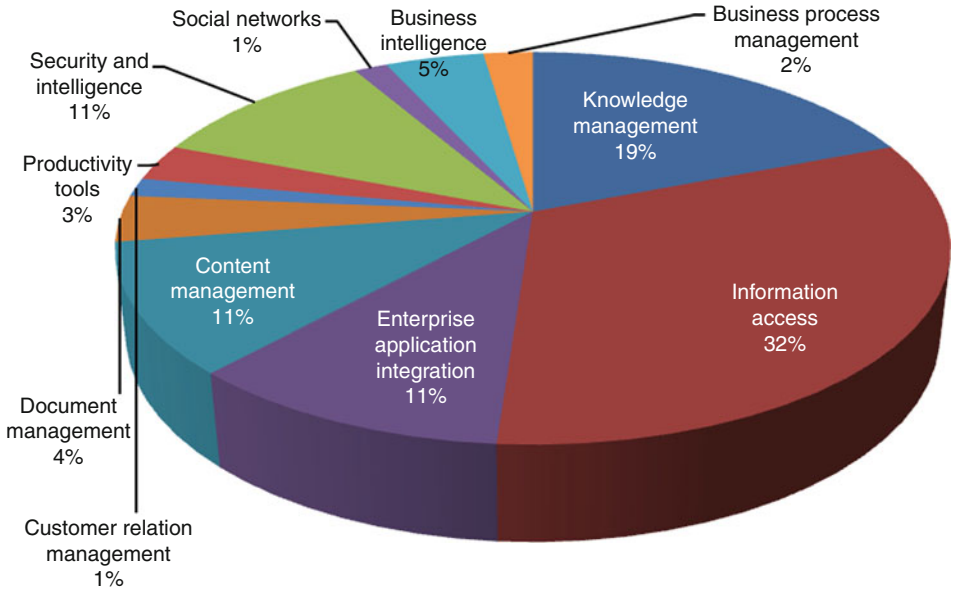


Fig. 15.9

Positioning of application areas on the semantic map reveals two clusters: the more conceptual areas and the “diverse context” areas



■ Fig. 15.10

Semantic offering according to area

perspective where “Search” is the largest category. The importance of Security and Intelligence is related to the defense sector.

While semantic technology is not yet in the mainstream market, analysis of the offering of 100 companies in the field shows that the majority of the solutions are in the “conceptual” space, that is, they offer solutions for problems that have a conceptual character and cannot be solved by traditional software. For the majority of the market, these problems are currently solved by people, not by software. A significant smaller number of offerings aim at problems related to integrating diverse business contexts. This is not surprising since “conceptual” offerings mostly apply to unstructured information that is usually not considered as enterprise-critical information, yet promises improvement in productivity. Data integration is usually applied on company-critical information stored in different databases, and before companies at large decide to use a new approach such as Semantic Technology to deal with their company-critical information, more experience and trust need to be gained.

This is actually a trend that can be projected on how and when semantic technology will be adopted by the mainstream market, which is the subject of the next section.

15.5 Adoption Horizons of Semantic Technology

Semantic technologies allow the addition of intelligence to traditional applications by managing the meaning of data. As has been explained, semantic technology is starting to have some impact in noncritical and internal environments in the scope of companies and organizations, where data and processes are controlled. As of 2009, some business

managers are testing its potential; other IT managers are testing its scalability and reliability. Semantic technologies allow for data or knowledge integration and if the current testing turns out to be successful, STE will be able to expand its scope both to more critical processes and to the external relationships of organizations with their environment (providers, customers, partners, etc.).

There are three possible phases in the adoption of semantic technology according to its penetration into organizations' structures. In the *first phase* some internal processes may use semantic technologies to enhance their capabilities. This technology is already in place and there is some track record in real environments [5].

In the *second phase*, semantic technologies allow for small ecosystem creation. Companies may integrate their value-chain processes using semantic technologies or vocabularies. Sector-based standards will boost the construction of dedicated semantic islands. There are some sectors (e.g., Health, Media, and Defense) that are already reaching this stage of STE penetration, even if the technology is still not mature and ready for massive deployment.

In the *third phase* full integration takes place of information systems using semantic technologies and vocabularies. Initiatives as Linked Data are some of the ongoing initiatives in this sense. More on linked data can be found in [▶ Semantic Annotation and Retrieval: Web of Data](#).

As of 2009, for the first phase (internal, noncritical), early adopters have deployed several semantic solutions and are now testing it in real settings. For the second phase (intercompany), the solutions are in prototypical stage and being evaluated by some early adopters. Those solutions might reach the mainstream market in the period 2012–2015. For the third phase (full integration, inter, intra, and worldwide), current solutions are still in research stage, and are not expected to reach market before 2015. The following sections will provide more detail of each of the phases along with two example use cases that illustrate typical applications.

15.5.1 First Phase: Short-Term Adoption of Semantic Technology

Short-term adoption refers to ongoing deployments of STE-based solutions in real environments. There are some market-ready solutions using semantic technology mainly dealing with companies' internal or noncritical processes such as knowledge management, innovation, or intranets. Also some companies are testing semantic technologies on their critical production systems, especially those with information integration and management requirements (e.g., pharmaceutical laboratories).

The increasing amount of knowledge-intensive work and more collaborative paradigms in companies are boosting the need for more intelligent information/knowledge retrieval and management systems. Companies are looking for new technologies and paradigms in order to foster employees' efficiency for knowledge acquisition and management. That is the case of communication, collaboration, and social networking technologies present in Web 2.0 that are being adopted within corporations and organizations.

There has arisen a new market for communication, collaboration, knowledge management, and social networking solutions for the enterprise. This new market, sometimes called Enterprise 2.0 or Social Software in the Workplace, is still an immature and heterogeneous concept, but according to some forecasts [6] may achieve \$4,6 billion by 2013, similar in size to the current Content Management Systems (CMS) market.

Enterprise 2.0 is a good example of how STE can penetrate the mainstream market in a short period with the following capabilities and tools:

- Semantic wikis for enterprise vocabulary and modeling
- STE underlying enterprise blogs publication
- Semantic document management
- Complex knowledge management applications
- Innovation and idea management
- Communication and collaboration tools

The usage of semantic technologies for internal processes as innovation is described in the business case above (🔗 [Table 15.1](#)).

Some pharmaceutical laboratories (e.g., Eli Lilly) use semantic technologies to speed up the drug development process as described above (🔗 [Table 15.2](#)).

15.5.2 Second Phase: Midterm Adoption of Semantic Technology

The upcoming networked economy is based on the ability of companies to transform information into knowledge and take profit from it. The agile and flexible reconfiguration of resources according to this knowledge has become key in the rapidly changing environment. In the midterm, semantic technology may enable the integration of several organizations in the same sector (vertical integration) or area (horizontal integration). Sectors or areas with an already existing controlled vocabulary or under strong regulation are better positioned for STE uptake.

Health, public administration, and defense are traditionally well-regulated sectors with a high penetration of controlled vocabularies and standards (🔗 [Table 15.3](#)).

15.5.3 Third Phase: Long-Term Adoption of Semantic Technology

At the long-term horizon, one can foresee the Semantic Web vision come true. Many of the semantic islands developed in previous phase may join in a common structure. The linked data [7] initiative has drastically pushed this vision into a reality. Despite scalability and maintenance issues, there already is available a common formalism for connecting semantically enabled domains. Companies in this phase will be able to extend their information systems to a general Web database.

Table 15.1

Use case on innovation process with STE support

Name	Collaborative innovation
Sector/Verticals	Any
Area/Horizontal	Knowledge Management
Target Customer	HR Director, Innovation Director, CEO
Decision Taker	<ul style="list-style-type: none"> ● Functional: HR Director, Innovation Director ● Technical: IT Manager ● Final User: Employee
Description	<p>In a customer-facing business, competitors often copy successful products and services within months after being introduced. This creates the need to improve existing products or processes, and introduce new ones quickly and frequently. Time to market is a critical factor, and technology plays an important role in speeding up the innovation process. The HR Department runs an innovation program through which they give financial rewards to employees who contribute the best ideas</p> <p>When an employee enters a new idea, the system analyzes the text and recognizes the relevant concepts from the perspective of the business. This happens in real time, enabling the user to be shown other ideas that contain the same concepts (not words!). It provides employees with a simple tool for checking whether their idea is actually new, or if it is a variation or complement of an existing idea</p> <p>The concepts are defined and related in an enterprise ontology, which includes products, channels, departments, clients, etc. In the same way, employees can search for ideas that contain relevant concepts, and concepts are highlighted in real time to provide feedback to the user</p> <p>Search and analysis can also be performed on dates, individual employees, departments, etc. Moreover, the system is able to give an explanation as to why it thinks certain ideas are similar by showing the semantic relations between the concepts in the new idea and existing ones</p>
Benefits	More efficient ideation process, tapping efficiently from all employees
Reference Users	Bankinter (Spain), Telefonica R&D (Spain), Repsol (Spain)

Currently, there are some ongoing initiatives in this sense, very much related to linked data (LD) standard in public (Table 15.4) and media (Table 15.5) sectors. Governments are publishing public data into the LD framework allowing third parties to construct intelligent applications on top (see, for example, <http://data.gov.uk>). More on the take-up of linked data in government can be found in eGovernment.

15.6 Conclusions and Summary

This chapter has reviewed relevant factors for bringing high-tech in general, and Semantic Technologies in particular, to the market place. The analysis is based on significant

■ **Table 15.2**

Use case on critical drug development process in a pharmaceutical lab

Name	Drug target assessment tool
Sector/Verticals	Health care: Pharmaceutical
Area/Horizontal	Knowledge Management
Target Customer	Scientists
Decision Taker	Technical: IT investment committee
Description	A drug target assessment tool that is part of a drug discovery infrastructure built using Microsoft's Composite Application Block technology. The implementation uses internally developed ontologies, and industry standard terminologies such as MESH. The ontologies are stored within the Oracle RDF Data Model, and linked to the diverse data sources that require integration
	The drug target assessment tool enables parallel assessment of candidate profiles across many scientific and business dimensions of interest. The interface allows scientists either to search directly for a given term and see all related data, or navigate to the term of interest through the ontology
	The results of queries display a set of entities as a graph to assist the user in visualizing and navigating among the relationships between the entities. This approach enables scientists to discover information as they navigate through available knowledge, rather than necessarily having to have a specific query in mind at the outset
	The tool gives researchers the ability to integrate diverse sources of data, view all data relating to entities of interest no matter where they are sourced, and the flexibility to incorporate additional unanticipated datasets in the research process
Benefits	<ul style="list-style-type: none"> ● Efficiency – reduced research wastage and opportunity costs ● Effectiveness – more accurate and flexible research process ● Cost – lower R&D cost per new drug
Reference Users	Eli Lilly

experience in working with Semantic Technologies and products (the vendor side), as well as on interviews with key business players (the client side). The main conclusions are:

1. In general, researchers and developers in the area are focusing more on the technology and less on the problems to be solved in terms of market needs, a situation that slows down the adoption rate of the technology.
2. There exists a gap between what vendors say they offer and what IT managers and CIOs understand they need in their business. That is, it is unclear for what types of companies Semantic Technologies provide a benefit. In order to bridge this gap, this work has identified the appropriate problem space in enterprises where it makes – business wise – sense to apply Semantic Technology today.

■ **Table 15.3**

Health care use case using STE

Name	Patient safety application for hospital management
Sector/ Verticals	Health care
Area/ Horizontal	Knowledge Management
Target Customer	Health systems
Decision Taker	Functional: Hospital Management Technical: IT Manager
Description	The health domain is traditionally one of the most advanced fields where the semantic technologies find a good breeding ground for testing and deploying compelling applications. This sector has defined communication standards (HL7, etc.), controlled vocabularies (GALEN, FMA, etc.), classification and reference norms (ICD, MESH, CPT, COSTAR, DSM IV, READ 3, etc.) and there even exist semantic models for the whole sector (SNOMED CT, UMLS, etc.). Making eHealth systems interoperable using common standard data formats and protocols facilitate a significant step forward in the achievement of a satisfactory health care: improving the care provided to patients, reducing medical errors, and saving human and economic costs. Experiences of fully automated local health systems with a lack of underlying standard for data exchange have shown that the gap between consumer expectations and the actual service delivery remains to be bridged
	Hospital patient management systems including STE help in the prevention of adverse events increasing patient safety. The semantic vocabulary and inference system alerts when possible mistakes may occur by establishing alarms or prevention maps for professionals reducing the possibility of human error
Benefits	Improve quality of health services. Prevent from legal costs
Reference Users	National Health Service, UK

3. Current offerings of semantic technology in the marketplace, as of 2009, are in large part related to problems that require the structuring of nonstructured information, and less to integration of companies' data.

The last part of the chapter identifies three phases through which semantic technology can enter the mainstream market.

In the rest of this book, the reader will find concrete experiences of applying Semantic Technologies to different business scenarios. It is an interesting exercise to contrast those business applications with the findings of this chapter. Applications that will be discussed include: Web search, eScience, Knowledge Management in large organizations, eBusiness, eGovernment, Multimedia Broadcasting and eCulture, and Semantic Web Services.

■ **Table 15.4**

Linked data use case for public administration

Name	UK Government linked data initiative
Sector/verticals	Public administration
Area/horizontal	NA
Target customer	Third-party companies for added-value application development
Decision taker	–
Description	Massive publication of public government data by the UK public administration (http://data.gov.uk). Data are published according to ST standards and can be used for intelligent application development. At this time (January 2009) it includes almost 3,000 datasets published by diverse global and local administrations
Benefits	
Reference users	UK Government

■ **Table 15.5**

Linked data use case for media industry

Name	BBC linked data dump
Sector/verticals	Media
Area/horizontal	NA
Target customer	Third-party companies for added-value application development
Decision taker	–
Description	BBC has published data about its programs (http://www.bbc.co.uk/programmes) and music (http://www.bbc.co.uk/music) in RDF format for further exploitation
Benefits	
Reference Users	BBC

15.7 Cross-References

- eBusiness
- eGovernment
- eScience
- Knowledge Management in Large Organizations
- Multimedia, Broadcasting and eCulture
- Social Semantic Web

Acknowledgment

The work reported in this chapter has been partially funded by the European Commission in the context of the Value-IT project.

Annex 1: STE Suppliers Company Listing

Company Selection

Companies were selected from the following sources.

STE-Related Web Pages

- Semantic Exchange: <http://www.semanticexchange.com>
- Semantic Report: <http://www.semanticreport.com>
- Semantic Website: <http://www.semanticweb.com>

EU-Funded Project and Commercial Reports

- Neon, Knowledge Web Deliverables
- Semantic Wave 2008 Report: Industry Roadmap to Web 3.0 and Multibillion Dollar Market Opportunities.: <http://www.project10x.com/index.php>

Conference Assistants

- International Semantic Web Conference (ISWC)
- European Semantic Web Conference (ESWC)
- Semantic Technology Conference (SemTech)
- European Semantic Technology Conference (ESTC)

Universities and public research centers are excluded from this study. Private research centers, R&D departments of big IT players have been included, since they usually transfer new technologies into the commercial portfolio (as was the case of Oracle, Yahoo!, SAP, etc.).

According to ones understanding of the STE, the following types of companies have been excluded from this listing:

- Natural Language Processing (NLP) technology based companies that may use the term of “semantics” for part of the NLP.

- Web development companies that may use microformats or RSS functionalities sometimes tagged as Semantic Technologies.

Company Description Fields

For each company included in this report, the following information is described:

- *Name*: Company name.
- *URL*: Corporate website. Almost all information included in this report was extracted from the corporate website of the companies.
- *Sectors*: According to company solutions and customers, the following sectors have been identified: Automotive, Defense, Public Sector, Energy, Health Care, Tourism, eCommerce, Retail, Financial, Media, Legal, Telco, Pharma, Academic and Educational, Advertising, IT.
- *Areas*: For each company, where information available, one or more semantic solution areas have been assigned related to their products or services: Knowledge Management, Information Access, Enterprise Application Integration, Content Management, Document Management, Customer Relation Management, Productivity Tools, Security and Intelligence, Social Networks, Business Intelligence, Business Process Management.
- *Technologies*: For each company, where information available, one or more semantic technologies have been assigned related to their products or services: Search, Document Tagging, Document Classification, Information Extraction, Semantic SOA, Modeling, Data Integration, Inference and Reasoning, Knowledge Publishing, Knowledge Storage, Hardware.

Supplier	URL	Areas	Technologies	Sectors
AdaptiveBlue	http://www.adaptiveblue.com	Productivity Tool		
Aduna	http://www.aduna-software.com	Information Access, Content Management	Knowledge Publishing, Modeling	Public Sector, Media
Agilense	http://www.agilense.com	Enterprise Application Integration	Data Integration, Knowledge Publishing	
Altova	http://www.altova.com		Modeling	
Amblit	http://www.amblit.com	Content Management	Modeling	

Supplier	URL	Areas	Technologies	Sectors
Ask	www.ask.com	Information Access	Search	
AskMeNow	http://www.askmenow.com	Information Access	Search, Data Integration	
Aspasia Systems	http://www.aspasia-systems.de	Knowledge Management		
Autonomy	http://www.autonomy.com	Information Access	Search	Media, Retail, Telco, Public Sector, Automotive, Finance, Defense, Legal, IT, Energy, Health, Pharma
BBN Technologies	http://www.bbn.com	Information Access, Content Management, Document Management	Search, Document Tagging, Information Extraction, Inference and Reasoning	Defense
Biowisdom	http://www.biowisdom.com	Knowledge Management, Information Access	Information Extration, Knowledge Publishing	Health
Boeing	http://www.boeing.com			
Cambridge Semantics	http://www.cambridgesemantics.com	Productivity Tools	Data Integration, Modeling	
Celtx	http://www.celtx.com	Modeling	Media	
CheckMI	http://www.checkmi.com			
Cognition	http://www.cognition.com	Information Access	Search, Document Tagging, Information Extraction	Health, Legal
Collexis	http://www.collexis.com	Knowledge Management, Social Networks	Search, Data Integration, Information Extraction, Knowledge Publishing	Health, Defense, Legal

Supplier	URL	Areas	Technologies	Sectors
Connotate	http://www.connotate.com	Business Intelligence, Information Access	Search, Knowledge Publishing	Finance, Legal, Defense, Media
Content Analyst	http://www.contentanalyst.com/	Information Access, Knowledge Management, Document Management		Legal, Education, Media
Contivo	http://www.contivo.com	Enterprise Application Integration, Business Process Management	Data Integration, SOA	
Convera	http://www.convera.com	Information Access	Search, Information Extraction	Media
Cougaar Software, Inc.	http://www.cougaarsoftware.com	Enterprise Application Integration	Data Integration	Defense
CyCorp	http://www.cyc.com	Security	Modeling, Reasoning	Finance
Digital Pebble	http://www.digitalpebble.com	Information Access	Information Extraction, Document Classification	
Digital Reasoning	http://www.digitalreasoning.com	Information Extraction	Document Tagging	Defense
DowJones	http://www.dowjones.com	Information Access, Knowledge management	Search, Modeling, Document Tagging, Knowledge Storage	Media, Financial, Energy, IT
EffectiveSoft	http://www.effectivesoft.com	Knowledge Management	Document Tagging, Document Classification, Information Extraction	
EMC Corporation	http://www.emc.com			

Supplier	URL	Areas	Technologies	Sectors
Empolis	http://www.empolis.com	Information Access, Content Management	Search, Document Tagging, Information Extraction	Telco, Manufacturing, Media, Public Sector
Endeca	www.endeca.com	Security and Intelligence, Information Access	Search, Document Tagging, Information Extraction	Retail, Financial, Automotive, Defense
Exalead	http://www.exalead.com	Information Access	Search, Knowledge Visualization	Media, Public Sector, Finance, Health, IT
Expert System	http://www.expertsystem.net	Knowledge Management, Security and Intelligence, Information Access	Search, Document Tagging, Information Extraction, Document Classification	Automotive, Telco, Defense, Tourism, Media
Expressor	http://www.expressor-software.com	Enterprise Application Integration	Data Integration	
Fortent	http://www.fortent.com	Business Intelligence, Security and Intelligence, Knowledge Management	Data Integration, Search, Reasoning,	Finance
Franz	http://www.franz.com/	Information Access	Modeling, Knowledge Storage	
Full Capture Solutions	http://www.fullcapture.com			Financial
Hakia	http://company.hakia.com	Information Access	Search	
Hewlett-Packard Company	www.hp.com			
IBM	http://www.ibm.com	Information Access	Search, Modeling, Semantic SOA, Knowledge Storage	

Supplier	URL	Areas	Technologies	Sectors
Image Matters	http://www.imagemattersllc.com	Knowledge Management, Information Access, Business Process Management, Enterprise Application Integration		
InfoSys	http://www.infosys.com	Information Access, Content Management	Search, Knowledge Publishing	
Intelius	http://www.intelius.com	Security and Intelligence		
Intellidimension	http://www.intellidimension.com		Knowledge Storage	
Intelligent Software Components	www.isoco.com	Knowledge Management, Business Intelligence, Content Management	Search, Information Extraction, Modeling, Knowledge Publishing	Finance, Public Sector, Tourism, Energy
Intellisophic	http://www.intellisophic.com			
invention-machine	www.invention-machine.com	Knowledge Management		Automotive, Energy, Health
IQSer	http://www.iqser.ch	Business Intelligence, Knowledge Management		
iSense	http://www.isense.net	Content Management	Information Extraction	Advertising
JustSystems	http://www.justsystems.com	Content Management, Enterprise Application Integration, Business Process Management		Defense, Financial, Public Sector
Kirix	http://www.kirix.com	Productivity Tools	Data Integration	

Supplier	URL	Areas	Technologies	Sectors
Knowledge-Based Systems, Inc.	http://www.kbsi.com	Knowledge Management	Information Extraction	Defense, Public Sector
Knowledge Foundations	http://www.knowledgefoundations.com	Knowledge Management	Information Extraction	
Lingway	http://www.lingway.com	Information Access, Knowledge Management	Search, Document Tagging	Health, IT
LinkSpace	http://www.linkspace.net	Information Access	Search	Defense
LucidMedia	http://www.lucidmedia.com	Content Management	Document Tagging, Document Classification	Advertising
Mandriva	http://www.mandriva.com	Information Access,	Data Integration, Search	IT
Metatomix	http://www.metatomix.com	Information Access, Knowledge Management	Data Integration	Legal, Financial
Microsoft	www.microsoft.com			
Mondeca	http://www.mondeca.com	Knowledge Management, Content Management, Information Access	Modeling, Search, Reasoning	Legal, Media, Defense
Motorola	http://www.motorola.com		Knowledge Publication	
Netbreeze	http://www.netbreeze.ch	Information Access	Document Tagging, Information Extraction, Search	Pharma, Financial
Nokia	http://www.nokia.com			
Norkom	http://www.norkom.com	Security		Financial
Ontopia	www.Ontopia.net		Modeling, Knowledge Publishing	

Supplier	URL	Areas	Technologies	Sectors
Ontoprise	http://www.ontoprise.de	Knowledge Management, Information Access, Content Management	Modeling, Knowledge Storage, Inference and Reasoning, Information Extraction	Automotive, Telco
Ontos	http://www.ontos.com	Business Intelligence, Document Management, Enterprise Application Integration	Document Tagging, Information Extraction, Semantic SOA	Finance, Legal
OntoText	http://www.ontotext.com	Information Access	Document Tagging, Modeling, Information Extraction, Language Understanding, Knowledge Storage	
Oracle	http://www.oracle.com	Enterprise Application Integration	Knowledge Storage, Data Integration	
Progress Software	http://www.progress.com	Enterprise Application Integration	Data integration	Defense, Finance, IT, Telco, Health, Media, Public Sector
Radar Networks	http://www.twine.com	Knowledge Management	Modeling, Data Integration	
Reinvent	http://www.reinvent.com	Content Management	Modeling, Publishing	Tourism
Reuse Company	http://www.reusecompany.com	Knowledge Management, Intelligence, Customer Relation Management	Modeling, Data Integration, Information Extraction	Public Sector, Energy
RiverGlass Inc.	http://www.riverglassinc.com	Knowledge Management, Intelligence	Search, Document Tagging, Information Extraction	Defense

Supplier	URL	Areas	Technologies	Sectors
Saltlux	http://www.saltlux.com	Information Access	Search, Information Extraction, Document Tagging	Telco, Legal, Public Sector
Sandpiper Software	http://www.sandsoft.com		Modeling, Data Integration	
SAP	http://www.sap.com	Enterprise Application Integration	Data Integration, Semantic SOA	
SchemaLogic	http://www.schemalogic.com	Information Access, Document Management	Search, Modeling	IT, Media, Defense
Semantech	http://www.semantech-inc.com	Enterprise Application Integration		
Semantic Arts	http://www.semanticarts.com	Enterprise Application Integration	Semantic SOA	Public Sector, Health Care
Semantic Research	http://www.semanticresearch.com	Intelligence, Knowledge Management	Knowledge Publishing, Modeling, Knowledge Storage	Education, Defense
Semantic System	http://www.semanticsystem.com		Hardware	
Siderean Software Inc.	http://www.siderean.com	Content Management, Knowledge Management, Information Access	Data Integration, Modeling, Knowledge Publishing	
Siemens	http://www.siemens.com		Modeling, Data integration	Health, Energy
SmartLogic	http://www.smartlogic.com	Information Access, Knowledge Management	Modeling, Document Classification, Knowledge Publishing	Public Sector, Media, Finance
Talis	http://www.talis.com	Document Management		Academic
TeraDact	http://teradact.com/index.htm			

Supplier	URL	Areas	Technologies	Sectors
Teradata	http://www.teradata.com	Enterprise Application Integration, Customer Relation Management	Data Integration	
Textwise	http://www.textwise.com	Information Access	Document Tagging, Information Extraction	
Thales	http://www.thalesgroup.com	Intelligence and Security		Defense, Health, Public Sector
Thetus	http://www.thetus.com	Knowledge Management, Information Access	Search, Data Integration, Modeling, Knowledge Publishing	Defense
Thomson Reuters	http://www.thomsonreuters.com	Intelligence, Information Access,		
TopQuadrant	http://www.topquadrant.com	Information Access	Modeling, Data Integration	Defense, Public Sector, Pharma, Automotive
TrueKnowledge	http://www.trueknowledge.com	Information Access	Search	
WAND	http://www.wandinc.com	Information Access	Modeling	eCommerce
Whatever	http://www.whatever-company.com	Knowledge Management, Social Networks	Search	
Ximetrix	http://www.ximetrix.com	Content Management	Search, Document Tagging, Knowledge Publishing	Public Sector
XSB	http://www.xsb.com	Information Access, Enterprise Application Integration, Business Intelligence	Data Integration, Information Extraction, Document Classification, Inference and Reasoning	

Supplier	URL	Areas	Technologies	Sectors
Yahoo!	www.yahoo.com	Information Access	Search, Document Tagging, Data integration	
Zemanta	http://www.zemanta.com	Productivity Tool	Content Tagging	Media
Zepheira	http://zepheira.com			Education
Zoominfo	http://www.zoominfo.com	Information Access	Data Integration, Search	

Annex 2: Company Listings by Sector, Technology, and Application Area

STE Companies by Sector

Companies are assigned one or multiple sectors according their offering and their customers. The following table shows the weight of some sectors of all selected companies, and then by address split into USA, EU, and others (► [Table 15.6](#)).

■ **Table 15.6**

STE companies by sector

Sector	Global		USA		EU		Others	
	110	100%	60	100%	40	100%	10	100%
Automotive	6	5%	4	7%	2	5%	0	0%
Defense	17	15%	13	22%	3	8%	1	10%
Public Sector	15	14%	5	8%	8	20%	2	20%
Energy	6	5%	3	5%	3	8%	0	0%
Health Care	11	10%	6	10%	5	13%	0	0%
Tourism	3	3%	0	0%	2	5%	1	10%
eCommerce	1	1%	1	2%	0	0%	0	0%
Retail	2	2%	2	3%	0	0%	0	0%
Financial	7	6%	4	7%	1	3%	2	20%
Media	14	13%	7	12%	7	18%	0	0%
Legal	9	8%	6	10%	1	3%	2	20%
Telco	6	5%	2	3%	3	8%	1	10%
Pharma	3	3%	2	3%	0	0%	1	10%
Academic and Educational	1	1%	0	0%	1	3%	0	0%
Advertising	2	2%	1	2%	1	3%	0	0%
IT	7	6%	4	7%	3	8%	0	0%

STE Companies by High-Level Technology

For each company a technology has been assigned according to the services or solutions it offers. Each company may have assigned more than one technology or any technology at all (► [Table 15.7](#)).

■ **Table 15.7**

STE companies by high-level technology

Technology	Global		USA		EU		Others	
	163	100%	94	100%	50	100%	19	100%
Search	32	20%	19	20%	10	20%	3	16%
Document Tagging	19	12%	9	10%	6	12%	4	21%
Document Classification	6	4%	2	2%	3	6%	1	5%
Information Extraction	22	13%	10	11%	8	16%	4	21%
Semantic SOA	5	3%	3	3%	1	2%	1	5%
Modeling	25	15%	14	15%	9	18%	2	11%
Data Integration	25	15%	21	22%	4	8%	0	0%
Inference and Reasoning	6	4%	4	4%	2	4%	0	0%
Knowledge Publishing	14	9%	6	6%	5	10%	3	16%
Knowledge Storage	8	5%	6	6%	2	4%	0	0%
Hardware	1	1%	0	0%	0	0%	1	5%

■ **Table 15.8**

STE companies by area

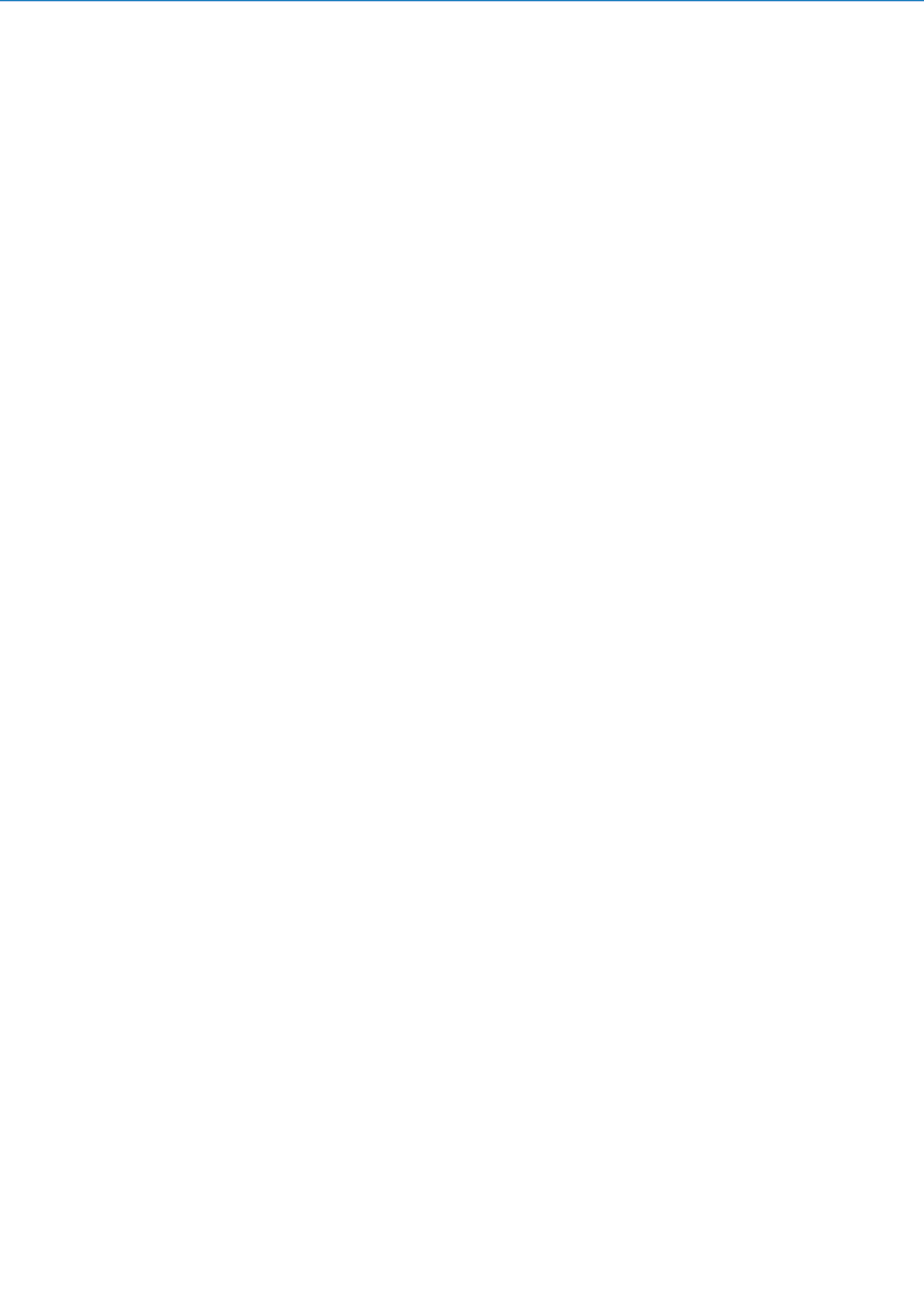
Area	Global		USA		EU		Others	
	132	100%	76	100%	40	100%	16	100%
Knowledge Management	25	19%	14	18%	9	23%	2	13%
Information Access	42	32%	26	34%	13	33%	3	19%
Enterprise Application Integration	14	11%	11	14%	1	3%	2	13%
Content Management	14	11%	4	5%	7	18%	3	19%
Document Management	5	4%	3	4%	1	3%	1	6%
Customer Relation Management	2	2%	1	1%	1	3%	0	0%
Productivity Tools	4	3%	3	4%	1	3%	0	0%
Security and Intelligence	14	11%	8	11%	4	10%	2	13%
Social Networks	2	2%	1	1%	1	3%	0	0%
Business Intelligence	6	5%	3	4%	1	3%	2	13%
Business Process Management	3	2%	2	3%	0	0%	1	6%

STE Companies by Area

◆ Table 15.8

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web, a new form of web content that is meaningful to computers will unleash a revolution of new possibilities. *Sci. Am. Mag.* **284**, 35–43 (2001)
2. Moore, G.: *Crossing the Chasm: Marketing and Selling High-Tech Products to Mainstream Customers*. HarperCollins, New York (1999)
3. Chesbrough, H.W.: *Open Innovation: The New Imperative for Creating and Profiting from Technology*. Harvard Business School, Boston (2003)
4. Value IT Consortium. Establishing dynamic link between research and business environments. <http://www.value-it.eu> (2010)
5. W3C. Semantic web case studies and use cases. <http://www.w3.org/2001/sw/sweo/public/UseCases/> (2010)
6. Young, G.O.: *Global Enterprise Web 2.0 Market Forecast: 2007–2013*. Forrester Research, Cambridge (2008)
7. Heath, T.: Linked data-connect distributed data across the web. <http://linkeddata.org/> (2010)



16 Semantic Web Search Engines

*Mathieu d'Aquin*¹ · *Li Ding*² · *Enrico Motta*¹

¹The Open University, Milton Keynes, UK

²Rensselaer Polytechnic Institute, Troy, NY, USA

16.1	<i>Scientific and Technical Overview</i>	661
16.1.1	Challenges	662
16.1.2	Related Systems	663
16.1.3	Abstract Specification	664
16.1.4	Case Study 1: Swoogle	667
16.1.4.1	Architecture	668
16.1.4.2	Crawling	668
16.1.4.3	Indexing	670
16.1.4.4	Ranking	670
16.1.4.5	Retrieval	671
16.1.4.6	Archive	672
16.1.5	Case Study 2: Watson	672
16.1.5.1	Architecture	673
16.1.5.2	Collecting Semantic Content: Crawling the Semantic Web	673
16.1.5.3	Analyzing Semantic Content: Validation, Indexing, and Metadata Generation	674
16.1.5.4	Web Interface: Search, Navigation, and Exploration	676
16.1.5.5	The Watson API	678
16.2	<i>Example Applications: Semantic Web Search Engines in Action</i>	680
16.2.1	Semantic Web Search Engines as Development Platforms	680
16.2.1.1	Scarlet: Relation Discovery	680
16.2.1.2	Swoogle Ontology Dictionary	680
16.2.1.3	Sig.Ma	681
16.2.1.4	The Watson Plug-In for Knowledge Reuse	682
16.2.1.5	Swoogle-Based Triple Shop	683
16.2.1.6	Evolve: Ontology Evolution Using Background Knowledge	685
16.2.1.7	Wahoo/Gowgle: Query Expansion	685
16.2.1.8	SWAML	686
16.2.1.9	PowerAqua: Question Answering	686
16.2.1.10	PowerMagpie: Semantic Browsing	687
16.2.1.11	FLOR: Folksonomy Ontology Enrichment	687

16.2.1.12	The Watson Synonym Service	688
16.2.2	Semantic Web Search Engines as Research Platforms	689
16.2.2.1	Swoogle-Based Semantic Web Statistics	689
16.2.2.2	Characterizing Knowledge on the Web with Watson	690
16.2.2.3	Measuring Ontology Agreement and Disagreement in Watson	692
16.3	<i>Related Resources</i>	696
16.4	<i>Conclusion and Future Directions</i>	697
16.5	<i>Cross-References</i>	698

Abstract: The last couple of years have seen an increasing growth in the amount of Semantic Web data made available, and exploitable, on the Web. Compared to the Web, one unique feature of the Semantic Web is its friendly interface with software programs. In order to better serve human users with software programs, supporting infrastructures for finding and selecting the distributed online Semantic Web data are needed. A number of Semantic Web search engines have emerged recently. These systems are based on different design principles and provide different levels of support for users and/or applications. In this chapter, a survey of these Semantic Web search engines is presented, together with the detailed description of the design of two prominent systems: Swoogle and Watson. The way these systems are used to enable domain applications and support cutting-edge research on Semantic Web technologies is also discussed. In particular, this chapter includes examples of a new generation of semantic applications that, thanks to Semantic Web search engines, exploit online knowledge at runtime, without the need for laborious acquisition in specific domains. In addition, through collecting large amounts of semantic content online, Semantic Web search engines such as Watson and Swoogle allow researchers to better understand how knowledge is formally published online and how Semantic Web technologies are used. In other terms, by mining the collected semantic documents, it becomes possible to get an overview and explore the Semantic Web landscape today.

The first section below (➤ Sect. 16.1) presents a general overview of the area, including the main challenges, related systems, as well as an abstract specification of what is called Semantic Web search engines. It also includes a detailed overview of the two systems more specifically considered as case studies, Swoogle (➤ Sect. 16.1.4) and Watson (➤ Sect. 16.1.5). ➤ Section 16.2 shows how these systems are currently being used and applied, both as development platforms to make possible the realization of applications exploiting Semantic Web content (➤ Sect. 16.2.1), and as research platforms, allowing one to better understand the content of the Semantic Web, how knowledge is published online and how it is structured. Finally, ➤ Sect. 16.3 briefly introduces other resources to be considered in the area of Semantic Web search engines, and ➤ Sect. 16.4 concludes the chapter.

16.1 Scientific and Technical Overview

In the early years, the deployment of the Semantic Web has been hindered by a dilemma on ontology reuse: ontology developers wanted others to adopt ontologies they created but they seldom adopted the ontologies created by others. Ontologies and knowledge bases were generally tailored to fit specific domain applications, which were rarely open to multiple, external ontologies and did not have to tackle the issues related to data integration, ontology coevolution, etc. This situation could be attributed to a number of reasons such as the existence of alternative standards, formalisms and languages (e.g., RDF and Conceptual Graphs [1]), the difficulties in integrating knowledge from different sources (e.g., DAML time ontology [2] and SOUPA [3] time ontology), and most importantly, to the limited support for finding and selecting reusable knowledge on the Web.

With the great efforts on standardization (see, e.g., RDF [4], OWL [5], URIs for the Semantic Web [6], SPARQL [7]), the fast-growing linked data (see, e.g., [8, 9]), and the advance of technologies such as robust storage, querying, and manipulation systems, the Semantic Web is now deemed as a huge success, at least according to one particular measure – availability: vast amounts of Semantic Web data are now directly made accessible from the Web for applications to reuse; SPARQL endpoints have been deployed all over the world to host particular datasets in specific domains; and more and more datasets encoded in relatively confined ontologies are now getting linked to the linked data cloud, which is leading to the ultimate “Web of Data” vision of the Semantic Web.

As a consequence, new challenges emerge surrounding the data accessibility issues: How to make the huge amount of Semantic Web data and data services published on the Web accessible by Web users, especially those unexpected consumers who are not familiar with the published datasets? How to facilitate applications access and integrate distributed Semantic Web data at web-scale? What kind of applications and research can be conducted with access to all the Semantic Web data published on the Web? What sort of support is needed by these applications for effectively using such knowledge? Semantic Web search engines, therefore, are developed to address these issues.

16.1.1 Challenges

The core challenges surrounding data accessibility can be summarized as making ontologies and data distributed on the Web accessible by intelligent applications to effectively take advantage of the Semantic Web as a distributed and interlinked knowledge base. Of course, more specific challenges emerge from this goal:

Heterogeneity: Despite the effort in standardizing technologies, at a higher level, the Semantic Web is characterized by heterogeneity along several dimensions, such as ontology quality, complexity, modeling, and views. A nontrivial effort is necessary to provide a homogeneous view and homogeneous access mechanisms to such heterogeneous information.

Scalability: With its millions of documents and billions of triples, the Semantic Web is already well beyond the size of any existing knowledge base in any semantic application. Although applications and users of the Semantic Web typically focus on a subset of what is available, efficient access mechanisms are required, and a shift is necessary for applications to locate and process the relevant information. Moreover, the open nature and the current rate of growth of the Semantic Web make it unrealistic to keep all Semantic Web data in a completely centralized manner; therefore, it is always desired to have relevant Semantic Web documents filtered before use.

Quality: Perfect quality cannot be assumed even in the absence of parser failure or semantic inconsistency. Information on the Semantic Web originates from many different sources and therefore varies considerably in quality. Trust becomes a key

factor in using the Semantic Web and increasing amount of interests have been projected on ranking both the importance of Semantic Web resources, and the level of confidence with which these resources can be used.

16.1.2 Related Systems

Several Semantic Web search engines have recently appeared (see [Sect. 16.3](#)) with the aim to tackle the above challenges. Aiming at an infrastructure for providing an effective access to Semantic Web data, Semantic Web search engines share the following characteristics: (1) They can scale up to web-scale, that is, they are to provide an effective index for all known Semantic Web data published on the Web. Instead of directly answering queries to Semantic Web data, they use their global index to filter the relevant dataset to be used to answer queries. (2) They can provide ranking to help users deal with alternative data, and thus better assist the selection of ontologies or semantic documents of different qualities on the Web. (3) They can provide advanced “semantic-based” services to human users and computer applications, and thus enable computer-assisted search-then-query processes. In this way, they help human users better leverage the automated processing of information to conduct intelligent filtering and integration tasks.

In order to clarify the scope of Semantic Web search engines, in the following are briefly presented other categories of systems that partly share the goal of data accessibility with Semantic Web search engines.

Database systems and knowledge-based systems generally focus on answering questions using well-structured knowledge stored in closed databases or knowledge bases. The typical input is a query encoded in a formal language, such as KIF (<http://www-ksl.stanford.edu/knowledge-sharing/kif/>) or SQL (<http://en.wikipedia.org/wiki/SQL>); the typical output are variable bindings that answer the query using the stored data or knowledge. Recent advances on natural language processing (NLP) technologies such as Controlled Natural Language [10] have been used to help users in composing structured queries using natural language. Existing triple store systems can be classified as database systems or knowledge base systems depending on whether inference (e.g., RDFS or OWL inference) is executed to answer queries on data encoded in RDF graph and the corresponding ontologies. SPARQL queries are used as data access interface and the query results with bindings to RDF resources and triples are typical output. It is notable that SPARQL by itself does not encode any inference requirements, and most triple stores provide SPARQL interface with various back-end inference capability on RDFS semantics and OWL semantics. Triples store queries, database queries, and knowledge base queries share similar focus on a limited scope of data even though they could be in huge volumes, and the results are expected to be complete and sound.

Web Search and Semantic Search focus on filtering relevant text documents. Using keyword-based queries, they return documents that, in the basic case of Web search, simply contain the keywords. Semantic Search extends this conventional scenario by adding some semantic components to better exploit the intended meaning of the keywords, as well as the

semantic content of the documents being searched. There have been a number of systems implementing a variety of tasks that relate to Semantic Search. For example, computer-assisted semantic query expansion based on latent semantic analysis is used to improve search results. In this way, Cuil.com presents follow-up drill down links by understanding what users want (note that on September 17, 2010 the Cuil servers were permanently taken offline). Semantic tags and annotations can also be attached to a document to better identify its content. These semantic indexes for documents can be manually entered or automatically extracted from the semantic analysis of the documents (see, e.g., PowerSet.com).

In comparison, Semantic Web search engines focus on Semantic Web data published on the open Web. They are specialized to search for documents or objects published on the Web using standard Semantic Web languages. They do not try to answer the queries directly like triple stores, but return relevant data to answer queries. Generally, they take keywords with simple constraints (e.g., restricting to particular types of entities) as input, although more formal query and exploration mechanisms are often available. Their goal is to provide a simple access point for these data, acting like classical search engines do for Web documents, but retrieving and delivering the URLs or materializations of the relevant Semantic Web data, and providing a basic Web-service infrastructure for applications to make use of these data and knowledge.

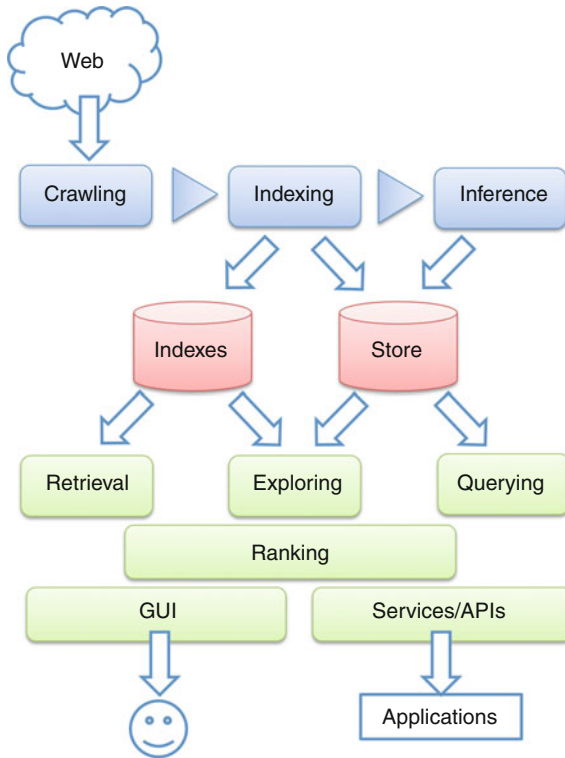
16.1.3 Abstract Specification

There are a number of initiatives that have emerged from the need for efficient, robust, and scalable Semantic Web search engines. While all these systems take different perspectives on the task of Semantic Web search, have different focuses, and are based on different assumptions, there exists a common ground that relates them to each other. This section intends to give the specification of this common base for Semantic Web search engines.

A Semantic Web search engine is a system that collects, indexes, and analyzes Semantic Web documents to provide search and querying mechanisms. Semantic Web documents are documents containing information encoded using standard Semantic Web languages such as RDF, RDFS, and OWL.

➤ *Figure 16.1* gives a general overview of the common activities of Semantic Web search engines. Not all of these components are present in all the search engines. For example, some systems rely only on manual submissions of semantic documents and do not use a crawler. However, this provides a general framework to which existing systems can be related and distinguished according to the way they implement the included components.

Crawling. Crawling is an essential task for systems with ambition to provide access to the whole set of semantic documents available on the Web. To some extent, crawling here is very similar to crawling for Web documents. However, the links that are followed by the crawler can be different (imports, explicit references through namespaces, etc.) Also, crawlers in Semantic Web search engines can exploit different sources of information to locate documents. For example, specific extensions of the sitemap mechanism have been developed (<http://sw.deri.org/2007/07/sitemapextension/>), as well as formats to describe

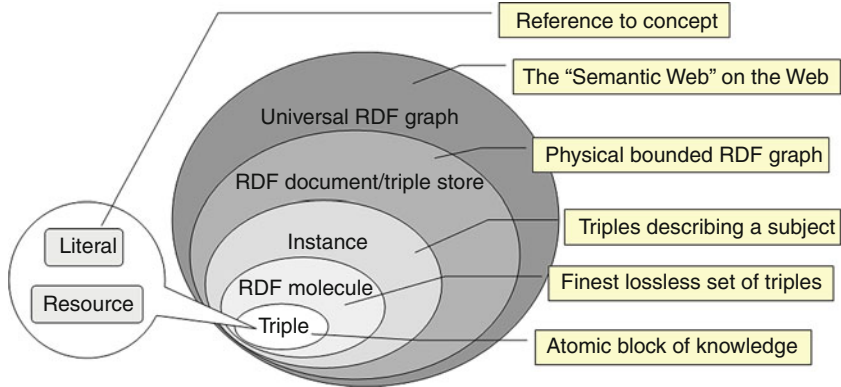


■ Fig. 16.1

High-level view of the activities of a Semantic Web search engine

semantic datasets online (<http://semanticweb.org/wiki/VoiD>). A system called *PingTheSemanticWeb.com* is dedicated to alerting Semantic Web crawlers of the appearances and updates of semantic documents online. In addition to the task of locating semantic documents, many refinements can be considered, including the necessary activity of *re-crawling* for evolving documents, of *meta-crawling* using other Web search engines, as well as the management of the overall crawling process, for example, using a pipelined approach [11]. Finally, the crawler is the part of a search engine where it is decided what should count as a semantic document, and what should be the boundaries of such a document. Indeed, Semantic Web data can be searched at different levels of granularity (see ▶ Fig. 16.2), ranging from the universal graph of all RDF data on the Web to a single RDF triple or even the constituent terms such as a URI. Also, some search engines may be more relaxed than others with respect to what can be included in their collection, filtering out, for example, RSS (RSS Feeds are arguably Semantic Web data because they are typically treated as XML data, as the related ontology barely use Semantic Web features.)

Indexing. One of the core elements of a Semantic Web search engine is its indexing process. Indeed, classical indexing mechanisms can be used to associate semantic documents to a set of terms, but most of the existing systems enhance such indexes for full-text



■ Fig. 16.2

The granularity levels range from the universal graph comprising all RDF data on the Web to individual triples and their constituent resources and literals

search with additional information such as metadata elements related to each document or indexes of the content of the documents (relations between entities) to allow for efficient querying and exploration mechanisms.

Inference. Inference can be used in a Semantic Web search engine to enhance the collected datasets and include inferred information. Heavy reasoning procedures might be used at indexing time (i.e., offline) as a one-time process, while lighter reasoning mechanisms (e.g., simple subclass transitive closure) might be realized at query time.

Ranking. As in Web search, the goal of ranking in Semantic Web search engines is to facilitate the selection of the most relevant information. However, the notion of relevance for semantic data can be more fuzzy and context dependent. Therefore, different systems adopt different approaches to the problem of ranking, from the use of simple measures originating from information retrieval [12], to more sophisticated metrics [13] and customizable ranking [14].

Retrieval. The data retrieval capabilities in different systems vary. The input ranges from keyword search to formal queries. Generally, results are URIs of Semantic Web documents, Semantic Web terms (i.e., classes and properties), and/or objects. Results can however be presented with certain amounts of additional associated metadata, and can be browsed in various ways.

Querying. While the search function is generally based on some form of keyword-based search, some systems can provide more formal ways to query the collection of documents they contain. A typical example is the use of SPARQL to allow users, but more importantly applications, to directly access the content of the documents, thus enabling their exploitation. Hence, some search engines may also play the role of global triple stores.

Navigation/Exploring. As mentioned above, Semantic Web search engines often provide browsable results, allowing the user to navigate the discovered documents (through the relations interlinking objects), to inspect the information attached to the documents

or to refine the query through query expansion mechanisms. These exploration mechanisms are also very useful to applications, as they provide specific points to drill down the relevant data, for example, an agent, once it has found a class `foaf:Person`, can further compose a precise query on finding FOAF documents by exploring all documents that declared at least one instance of `foaf:Person`.

Search interface. Most of the systems provide services to agents, allowing them to directly access the metadata and search results, in addition to a graphical user interface for human users. Different technologies might be used to deliver such interfaces and the level of features provided through these services can vary from simple search mechanisms to complete APIs for the exploration and exploitation of online semantic content.

The next two sections show how the abstract specification described above is instantiated in two of the most prominent systems currently deployed.

16.1.4 Case Study 1: Swoogle

In order to support consumers to find and surf the fast-growing Semantic Web data on the Web, Swoogle [15] has been designed and implemented to complement the conventional Web search engines. [Figure 16.3](#) illustrates a typical use of Swoogle in supporting web-scale Semantic Web data access. In this case, a software agent tries to answer queries using Semantic Web data on the Web via the following steps: (1) Swoogle crawls the Web for Semantic Web documents (SWDs) and Semantic Web terms (SWTs). It then builds an index for the harvested Semantic Web data and computes the corresponding rank. (2) The agent asks Swoogle's term search service using a keyword query "person" and is informed a suggested URI reference (URIref) `-foaf:Person`. (3) The agent then composes a SPARQL query using the retrieved URIrefs together with some known URIrefs. (4) The

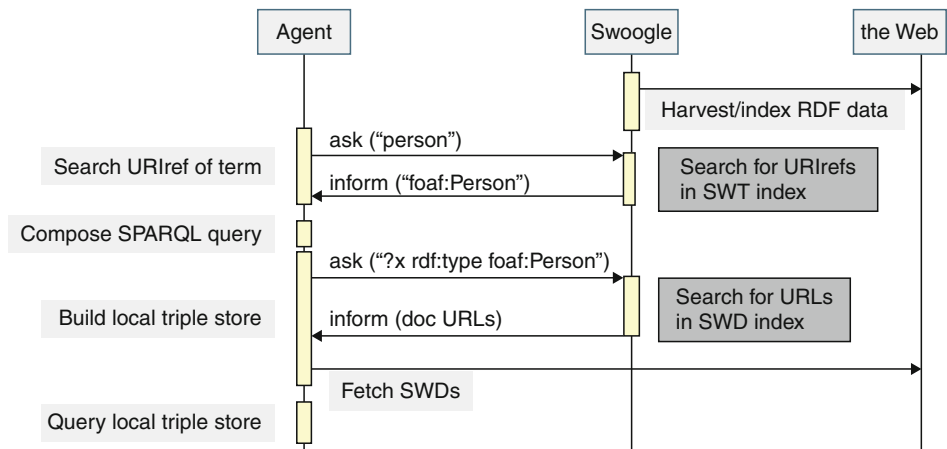


Fig. 16.3

A typical usage of Swoogle in web-scale Semantic Web data access

agent asks Swoogle's document search service for URLs of SWDs relevant to the SPARQL query. (5) The agent builds a local triple store by fetching the SWDs from the returned URLs. (6) The agent answers the SPARQL query using the integrated data in a local triple store.

16.1.4.1 Architecture

Similar to conventional Web search engines, Swoogle crawls the Web, builds indexes, computes ranks, and provides search services shown in [Fig. 16.4](#). Meanwhile, Swoogle is specialized for processing Semantic Web data on the Web. In what follows, several highlighted components in this architecture are elaborated.

16.1.4.2 Crawling

In order to effectively harvest SWDs on the Web, Swoogle uses a hybrid crawler that integrates several mechanisms for discovering and harvesting Semantic Web documents on the Web. [Figure 16.5](#) illustrates the conceptual workflow of the hybrid crawler, and the details are explained below.

1. *Bootstrapping*. Manually submitted URLs are used to bootstrap the discovery process by providing the seeding URLs for Google-based meta-crawling and bounded HTML crawling.
2. *Google-based Meta-crawling*. Meta-crawling [16] involves directly harvesting URLs from search engines without crawling the entire Web. Google is used for several reasons: (1) It has indexed the largest number of Web documents among existing

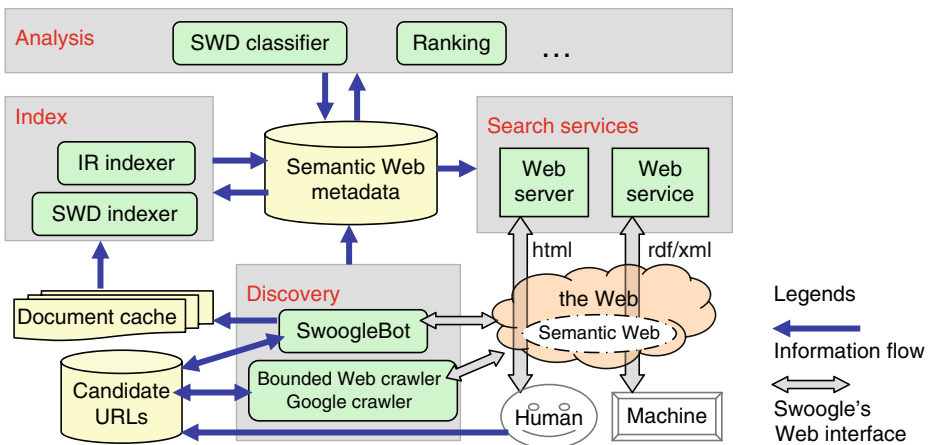
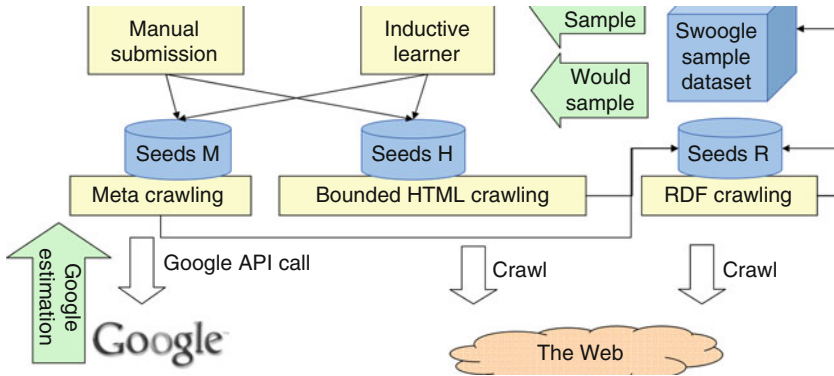


Fig. 16.4

The architecture of Swoogle



■ Fig. 16.5

The Swoogle system uses an adaptive Semantic Web harvesting framework with three different kinds of crawlers

Web search engines [17], (2) it does not filter Semantic Web documents out of search results, (3) it provides a Web API which is friendly to meta-crawlers, and most importantly, (4) it supports rich query constraints on both the text content and the URL of Web documents, namely “filetype,” “inurl,” and “site.” The crawler is provided with seeds from manual bootstrapping input and enriches the seeds using the *inductive learner* that selects “good” seeds from the harvested *Swoogle sample dataset*. A “good” seed is a Google query that is believed to contribute a high percentage of SWDs, for example, most URLs returned by the query *rdf filetype:rdf* are indeed SWDs.

3. *Bounded HTML crawling*. *HTML crawling* (i.e., conventional Web crawling) harvests Web documents by extracting and following hyperlinks, and is useful in harvesting clusters of SWDs on the Web. The *bounded HTML crawling* imposes some thresholds (e.g., crawling depth, maximum number of URLs to visit, and minimum percentage of SWD in visited URLs) to limit search space and ensure efficiency. For example, the crawler has harvested many PML documents (SWDs) that populate instances of the Proof Markup Language (PML) by a bounded HTML crawl starting at <http://iw.stanford.edu/proofs>. Again, manual submission and automated inductive learner are involved in collecting seeding URLs.
4. *RDF crawling*. The *RDF crawler* enhances conventional HTML crawling by adding RDF validation and semantic hyperlink extraction components. It also visits newly discovered URLs and periodically revisits pages to keep metadata up to date. For each URL, it tries to download the content of the Web page, and then parse an RDF graph from the document using popular RDF parsers (e.g., Jena). If successful, it generates document-level metadata for the SWD and also appends the newly discovered URLs that may link to SWDs to its to-visit list.
5. *Inductive learner and Swoogle sample dataset*. The sample dataset is obtained from the metadata of the SWDs confirmed by RDF crawling. Based on the features (e.g., URL, frequency of referred Semantic Web URIs, the source website) of harvested documents

and their labels (e.g., whether they are SWD, embedded SWD or non-SWD), an automated inductive learner is used to generate new seeds for Google-based meta-crawling and bounded HTML crawling.

The crawler schedules its methods according to the following strategies: (1) Semantic Web ontologies have the highest priority since ontologies are critical for users to encode and understand Semantic Web data, (2) Semantic Web documents in RDF/XML syntax have higher priorities than Web pages that embed Semantic Web data because the former usually contain more Semantic Web data, and (3) harvesting URLs from one website is delayed where more than 10,000 SWDs have already been found at the site (e.g., liveJournal) to avoid having the catalog dominated by SWDs from a few giant websites.


16.1.4.3 Indexing

The *Indexing* component analyzes the discovered SWDs and generates the bulk of Swoogle's metadata about the Semantic Web. The metadata not only characterize the features associated with individual SWDs and SWTs, but also track the relations among them, for example, "how SWDs use/define/populate a given SWT" and "how two SWTs are associated by instantiating 'rdfs:domain' relation" [12].

The annotation metadata of a URI include the namespace and local-name extracted from the terms URI; the literal description of the term from different SWDs. The annotation metadata of SWDs include metadata about itself (such as document URL and last modified time) and its content (such as terms being defined or populated and ontology documents being imported). Moreover, Swoogle maintains relational metadata that enable users to combine keyword search and hyperlink-based surfing to locate search targets.

16.1.4.4 Ranking

Google was one of the first Web search engines to order its search results based in part on a Web page's "popularity" as computed from the Web's graph structure. This idea has turned out to be enormously useful in practice and is applicable to Semantic Web search engines. However, Google's PageRank [18] algorithm, which is based on the "random surfer model," cannot be directly used in the Semantic Web for several reasons. URIs in a document are not merely hyperlinks but semantic symbols referring to classes, instances, ontology documents, normal Web resources, etc. Semantic Web surfing is not merely random hyperlink-based surfing but rational surfing that requires understanding the semantic content of documents.

In order to rank the popularity of Semantic Web documents, the rational surfing model is adopted: a rational surfer always recursively pursues the definition of classes and properties for complete understanding of a given RDF graph.  *Figure 16.6* illustrates the rational surfing behavior of a software agent, which unfolds as follows. The agent jumps randomly to one of the accessible SWDs with uniform probability. It either terminates

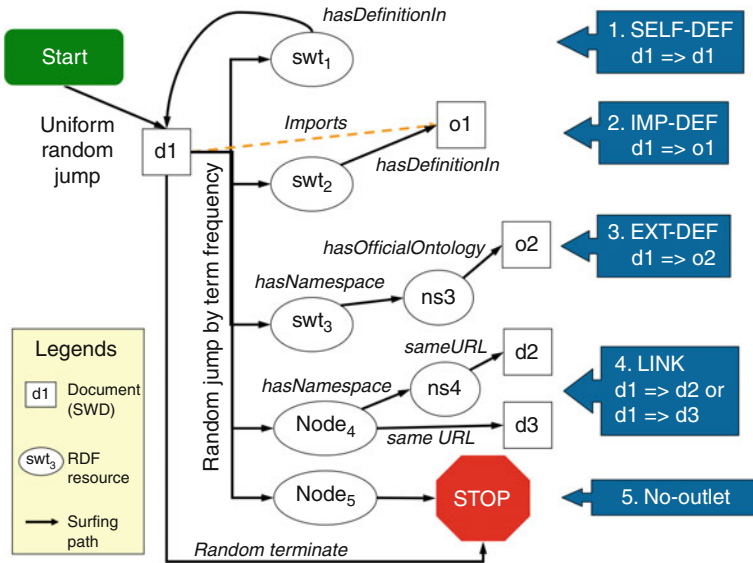


Fig. 16.6

Swoggles ranking algorithm is based on a “rational surfer model” that captures how a program might access links in processing Semantic Web documents

surfing with constant probability or chooses one RDF node in the RDF graph of the document, and the node is chosen based on its term frequency in the N-Triples version of the document. The agent either surfs to another document or terminates surfing based on the semantics of the chosen node. Paths 1 (SELF-DEF), 2 (IMP-DEF), and 3 (EXT-DEF) represent the agent pursuing a definition. If the node is not anonymous and is used as a class or property usage in the present document, the agent pursues further definition from the present document, the imported ontologies, or the ontology addressed by the namespace part of the node’s URI. Path 4 (LINK) shows the hyperlink-based surfing behavior: if the node is not anonymous and is not used as a class or property, the surfer follows the URL obtained from its URI or namespace to another Semantic Web document. Path 5 (No-outlet) includes all cases when no further surfing path starts from the present node, for example, the present node is literal or anonymous, or the present node’s URI links to a normal Web document.

16.1.4.5 Retrieval

The *retrieval* module provides search services to both human and software users using the indexed metadata. While queries to Web search engines return documents, the results of a Semantic Web search query can be at different levels of granularity: a Semantic Web document as well as a URI of Semantic Web terms (i.e., classes and properties). Currently, Swoogle provides two types of search services: (1) search for Semantic Web ontologies or all Semantic Web documents using keywords with additional query constraints, and

(2) search for Semantic Web terms using keywords with additional query constraints. Keywords are used to match the text parsed from the URI, labels, comments of a document, or a term. Additional query constraints can be used to filter the results using the indexed metadata, for example, only find SWTs defined as OWL class, only find SWTs defined using FOAF namespace, only find SWDs published at <http://inference-web.org>.

Nineteen REST Web service APIs are specially developed to support machine agents data access activities. A PHP-based website is built on top of the Swoogle APIs to support human users as well as to test the APIs. The service APIs are highlighted by demonstrating the enhanced Search and Navigation model [12].

16.1.4.6 Archive

Like most search engines, Swoogle keeps a cache of the publicly available Semantic Web documents it indexed. Furthermore, Swoogle goes beyond this in two ways. First, it also maintains a copy of each documents representation as a set of triples, a more useful form for programs and agents. Second, and more significantly, Swoogle maintains an archive of all of the current and old versions of each Semantic Web document in its index. The resulting Semantic Web Archive (http://swoogle.umbc.edu/index.php?option=com_swoogle_service&service=archive) can be used by researchers to study how ontologies evolve, to track the growth of documents containing RDF data or to investigate the natural life cycle of the Semantic Web.

16.1.5 Case Study 2: Watson

The research on Watson originates from the observation, and anticipation, that, more and more, the way intelligent applications will be developed will change due to the availability of a large-scale, distributed body of knowledge on the Web. The dynamic exploitation of this body of knowledge introduces new possibilities and challenges requiring novel infrastructures to support the implementation of a new generation of Semantic Web applications. New mechanisms are required to enable the development of such applications, exploring large-scale semantics [19, 20]:

Finding the relevant sources: The ability to locate dynamically the sources containing relevant semantic information is a prerequisite for applications that aim to leverage the use of online knowledge. This feature is important because, in such applications, the relevance of a particular resource to a problem-solving need cannot be judged at design time.

Selecting the appropriate knowledge: From the set of previously located semantic documents, the appropriate knowledge has to be selected based on application-dependent criteria, such as the quality of the data and its adequacy to the task at hand.

Exploiting heterogeneous knowledge sources: When reusing online semantic information, no assumption can be made on the ontological nature of the elements that are


manipulated. Hence the process needs to be generic enough so that it can make use of any online semantic resource. In addition, as in the case of the aforementioned tasks of finding and selecting semantic resources, this activity must also be carried out at runtime.

Combining ontologies and resources: It cannot be expected that one unique source of knowledge will provide all the required elements for a given application. Therefore, it is often necessary for next-generation Semantic Web applications to select and integrate partial fragments of knowledge from different sources, so that they can be exploited jointly.

Watson is a gateway to the Semantic Web: it collects, analyzes, and gives access to ontologies and semantic data available online. Its objective is to support the development of this new generation of Semantic Web applications that dynamically select, combine, and exploit the knowledge published on the Semantic Web.

16.1.5.1 Architecture

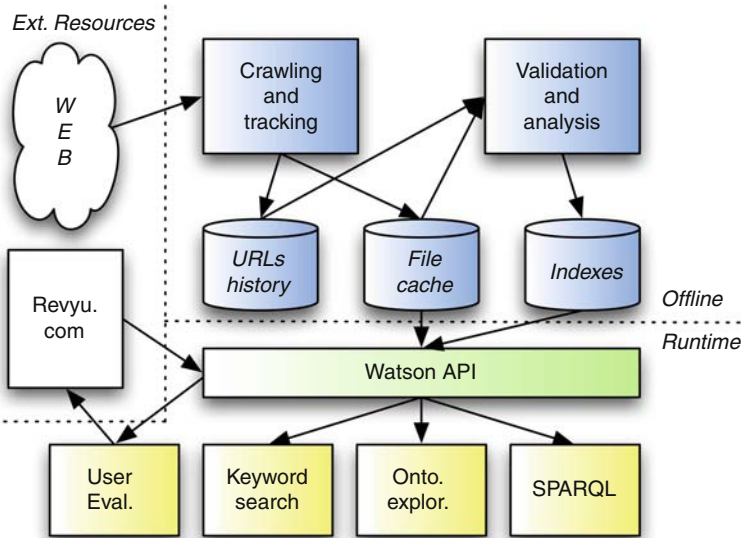
The role of a gateway to the Semantic Web is to provide an efficient access point to online ontologies and semantic data. Therefore, such a gateway realizes three main activities: (1) it collects the available semantic content on the Web, (2) analyzes it to extract useful metadata and indexes, and (3) implements efficient query facilities to access the data. While these three tasks are generally at the basis of any classical Web search engine, their implementation is rather different when dealing with semantic content as opposed to Web pages.

To realize these tasks, Watson is based on a number of components depicted in  Fig. 16.7, relying on existing, standard, and open technologies. Locations of existing semantic documents are first discovered through a *crawling and tracking* component, using in particular Heritrix, the Internet Archive's Crawler (<http://crawler.archive.org/>). The *Validation and Analysis component* is then used to create a sophisticated system of indexes for the discovered documents, using the Apache Lucene indexing system (<http://lucene.apache.org/>). Based on these indexes, a core API is deployed that provides all the functionalities to search, explore, and exploit the collected semantic documents. This API also links to the *Revyu.com* Semantic Web-based reviewing system to allow users to rate and publish reviews on ontologies.

16.1.5.2 Collecting Semantic Content: Crawling the Semantic Web

The goal of the crawling task in Watson is to discover locations of semantic documents and to collect them. Classical Web crawlers can be used, but they need to be adapted to take into account the fact that the crawler is not dealing only with Web pages, but also with semantic content.

Sources: Different sources are used by the crawler of Watson to discover ontologies and semantic data (Google, Swoogle, <http://pingthesemanticweb.com/>, etc.). Specialized crawlers were designed for these repositories, extracting potential locations by sending



■ Fig. 16.7

Overview of the Watson architecture

queries that are intended to be covered by a large number of ontologies. For example, the keyword search facility provided by Swoogle is exploited with queries containing terms from the most common words in the English language. Another crawler heuristically explores Web pages to discover new repositories and to locate documents written in certain ontology languages (e.g., by including “filetype:owl” in a query to Google). Finally, already collected semantic documents are frequently re-crawled, to discover evolutions of known semantic content or new elements at the same location.

Filters: Once located and retrieved, these documents are filtered to keep only the elements that characterize the Semantic Web. In particular, to keep only the documents that contain semantic data or ontologies, the crawler eliminates any document that cannot be parsed by Jena (<http://jena.sourceforge.net/>). In that way, only RDF-based documents are considered. Furthermore, a restriction exists, which imposes that all RDF-based semantic documents be collected with the exception of RSS. The reason to exclude these elements is that, even if they are described in RDF, RSS feeds represent semantically weak documents, relying on RDF Schema more as a way to describe a syntax than as an ontology language.

16.1.5.3 Analyzing Semantic Content: Validation, Indexing, and Metadata Generation

Many different elements of information are extracted from the collected semantic documents: information about the entities and literals they contain, about the employed

languages, about the relations with other documents, etc. This requires analyzing the content of the retrieved documents in order to extract relevant information (metadata) to be used by the search functionality of Watson.

Simple Metadata: Besides trivial information, like the labels and comments of ontologies, some of the metadata that are extracted from the collected ontologies influence the way Watson is designed. For instance, there are several ways to declare the URI of an ontology: as the namespace of the document, using the `xml:base` attribute, as the identifier of the ontology header, or even, if it is not declared, as the URL of the document. URIs are supposed to be unique identifiers in the scope of the Semantic Web. However, two ontologies that are intended to be different may declare the same URI [10, 21]. For these reasons, Watson uses internal identifiers that may differ from the URIs of the collected semantic documents. When communicating with users and applications, these identifiers are transformed into common, nonambiguous URIs.

Content: Another important step in the analysis of a semantic document is to characterize it in terms of its content. Watson extracts, exploits, and stores a large range of declared metadata or computed measures, like the employed languages/vocabularies (RDF, RDFS, OWL, DAML + OIL), information about the contained entities (classes, properties, individuals and literals), or measures concerning the richness of the knowledge contained in the document (e.g., the expressiveness of the employed language, the density of the class definitions, etc.). By combining these elements of information, Watson can decide whether or not a particular document should be treated as a semantically rich ontology. These elements are then stored and exploited to provide advanced, quality-related filtering, ranking, and analysis of the collected semantic content.

Relations between semantic documents: In the previous paragraphs, the analysis task was to extract metadata concerning one particular semantic document. In addition, a core aspect in the design of Watson concerns the exploitation of relations between semantic documents. The retrieved ontologies are inspected in order to extract information linking to other semantic documents. There are several semantic relations between ontologies that have to be followed (e.g., `owl:imports`, `rdfs:seeAlso`, `namespaces`, `dereferenceable URIs`). Besides providing useful information about the considered documents, the results of this task are also used to extract potential locations of other semantic documents to be crawled.

In addition to declared semantic relations like `owl:imports`, the aim is also to compute implicit relations that can be detected by comparing ontologies. Equivalence is one of the most obvious of these relations, which is nevertheless crucial to detect. Indeed, detecting duplicated knowledge ensures that redundant information is not stored and that duplicated results are not presented to the user. On the same basis, several other relations are considered relying on particular notions of similarity between ontologies (inclusion, extension, overlap, etc.). Combined with other information from the crawler (e.g., date of discovery, of modification), these relations make possible the study and characterization of the evolution of ontologies on the Web through their different versions.

16.1.5.4 Web Interface: Search, Navigation, and Exploration

Even if the first goal of Watson is to support semantic applications, it is important to provide Web interfaces that facilitate the access to ontologies for human users. Users may have different requirements and different levels of expertise concerning semantic technologies. For this reason, Watson provides different “perspectives,” from the most simple keyword search, to sophisticated queries using SPARQL (see [Fig. 16.8](#)). It can be accessed at the following address <http://watson.kmi.open.ac.uk/>.

Keyword search: The keyword search feature of Watson is similar in its use with usual Web or desktop search systems. The set of keywords entered by the user is matched to the local names, labels, comments, or literals of entities occurring in semantic documents. A list of matching ontologies is then displayed with, for each ontology, some information about it (language, size, etc.) and the list of entities matching each keyword. The search can also be restricted to consider only certain types of entities (classes, properties, individuals) or certain descriptors (labels, comments, local names, literals).

Ontology summaries: In order to facilitate the assessment and selection of ontologies by users, it is crucial to provide overviews of ontologies that are easy to read and understand, both at the level of the automatically extracted metadata about them, as well as at the level of their content. For each collected semantic document, Watson provides a page that summarizes essential information such as the size of the document (in bytes, triples, number of classes, properties, and individuals), the language used (OWL, RDF-S and DAML + OIL, as well as the underlying description logic), the links with other documents (through imports), and the reviews from users of Watson. Providing an appropriate overview of the content of an ontology or a semantic document is a difficult task. The complete graph of the content would not be really convenient for the user, and the natural language description contained in the comment about the ontology is rarely present, and generally not precise enough to help understanding the information formalized within this ontology. In other terms, there is a need to summarize ontologies, providing concise descriptions of the most important elements they contain. Peroni et al. [22] present a method to automatically extract the key concepts of an ontology using a variety of dimensions. The key concepts of an ontology are the concepts that are considered the best descriptors of the ontology by human users. In Watson, this work is used to generate small graphs, showing the six first key concepts of each ontology and an abstract representation of the existing relations between these concepts (see the example in [Fig. 16.9](#)). These visual summaries of ontologies provide a convenient way to obtain a quick overview of the considered ontology, which can be completed by a more precise and detailed exploration of the ontology if necessary.

Ontology exploration: One principle applied to the Watson interface is that every URI is clickable. A URI displayed in the result of the search is a link to a page giving the details of either the corresponding ontology or a particular entity. Since these descriptions also show relations to other elements, this allows the user to navigate among entities and ontologies. It is therefore possible to explore the content of ontologies, navigating through the relations between entities as well as to inspect ontologies and their metadata.

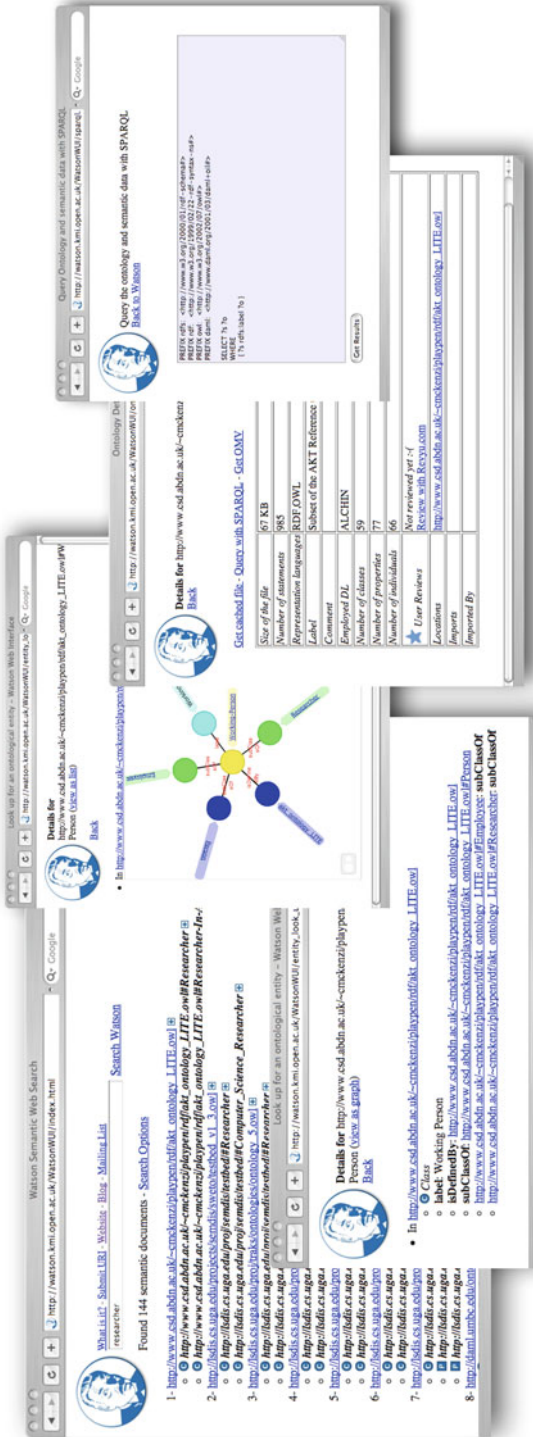
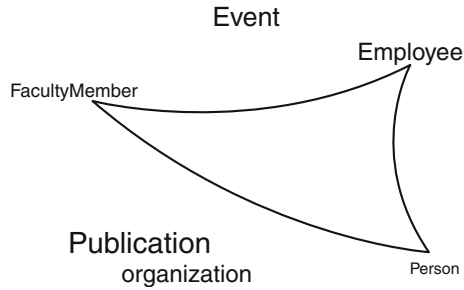


Fig. 16.8 Overview of the Watson Web interface



■ Fig. 16.9

Key concept–based visual summary of the ontology <http://swrc.ontoware.org/ontology/portal>

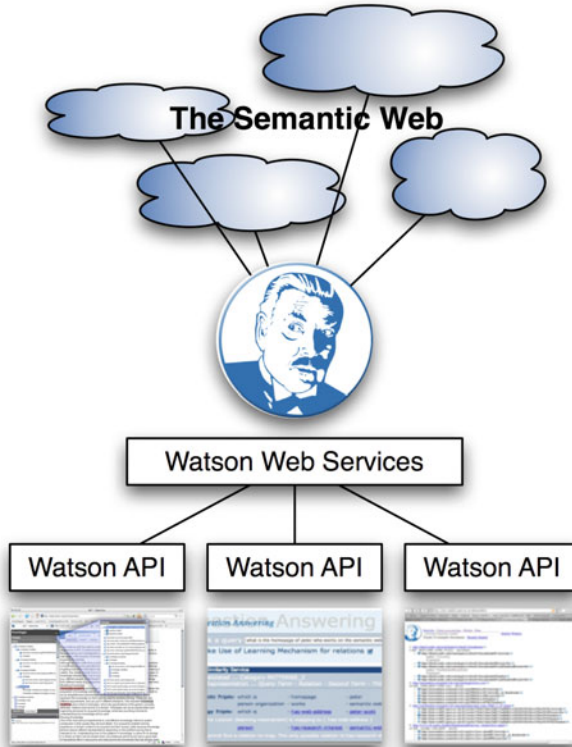
SPARQL. A SPARQL endpoint has been deployed on the Watson server and is customizable to the semantic document to be queried. A simple interface allows one to enter a SPARQL query and to execute it on the selected semantic document. This feature can be seen as the last step of a chain of selection and access tasks using the Watson Web interface. Indeed, keyword search and ontology exploration allow the user to select the appropriate semantic document to be queried. The next step is to extend this feature to be able to query not only one semantic document at a time, but also to automatically retrieve the semantic data useful for answering the query.

16.1.5.5 The Watson API

The core components of Watson are the services and API it provides to support the development of next-generation Semantic Web applications (see ▶ Fig. 16.10). Indeed, Watson deploys a number of Web Services and a corresponding API allowing applications to:

- Find Semantic Web documents through sophisticated keyword-based search, allowing applications to specify queries according to a number of parameters (type of entities, level of matching of the keywords, etc.)
- Retrieve metadata about these documents, for example, size, language, label, logical complexity, etc.
- Find specific entities (classes, properties, individuals) within a document
- Inspect the content of a document, that is, the semantic description of the entities it contains
- Apply SPARQL queries to Semantic Web documents

In sum, Watson’s API provides a number of advantages. In Watson, it is considered that any piece of information that has been collected should be made available, so that applications are provided with as much information as possible. Also, the comprehensive set of functionalities exposed by the API allows any application to use online semantic



■ Fig. 16.10

Using the Watson API to build Semantic Web applications

data in a lightweight fashion, without even having to download the corresponding semantic documents. The content of a semantic document is processed and indexed by Watson so that it can be accessed by applications at runtime, without requiring sophisticated mechanisms and large resources.

The combination of mechanisms for searching semantic documents (keyword search), retrieving metadata about these documents and querying their content (e.g., through SPARQL) provides all the necessary elements for applications to select and exploit online semantic resources. Moreover, the Watson Web Services and API are in constant evolution to support the requirements of novel applications. In particular, an initial set of measures, which evaluate the complexity and richness of ontologies, is currently being used for ranking. A more flexible framework combining both automatic metrics for ontology evaluation and user evaluation is being developed to allow for a more customizable selection mechanism. Another important direction concerns the detection of semantic relations between ontologies to support their combination. Indeed, while a simple duplicate detection mechanism is already in place, more advanced mechanisms need to be considered to efficiently discover fine-grained relations such as extension, version, or compatibility.

16.2 Example Applications: Semantic Web Search Engines in Action

16.2.1 Semantic Web Search Engines as Development Platforms

A number of applications relying on Watson, Swoogle, and other Semantic Web search engines have been developed and provide demonstrators of the possibilities offered by exploiting the Semantic Web. This section describes a few selected applications in different categories (services, ontology and semantic data management tools, end-user applications) with the aim of providing an overview of the variety of tasks that can be achieved nowadays with the Semantic Web. More details can be found in [19, 23].

16.2.1.1 Scarlet: Relation Discovery

Scarlet (<http://scarlet.open.ac.uk/>) follows the paradigm of automatically selecting and exploring online ontologies to discover relations between two given concepts. For example, when relating two concepts labeled *Researcher* and *AcademicStaff*, Scarlet, using Watson, (1) identifies (at runtime) online ontologies that can provide information about how these two concepts interrelate and then (2) combines this information to infer their relation. Two increasingly sophisticated strategies were investigated to discover and exploit online ontologies for relation discovery. The first strategy, S1, derives a relation between two concepts if this relation is defined within a single online ontology, for example, stating that *Researcher* \sqsubseteq *AcademicStaff*. The second strategy, S2, addresses those cases in which no single online ontology states the relation between the two concepts, by combining relevant information which is spread over two or more ontologies, for example, that *Researcher* \sqsubseteq *ResearchStaff* in one ontology and that *ResearchStaff* \sqsubseteq *AcademicStaff* in another. To support this functionality, Scarlet relies on Watson to access online ontologies.

Scarlet originates from the design of an ontology matcher that exploits the Semantic Web as a source of background knowledge to discover semantic relations (mappings) between the elements of two ontologies. This matcher was evaluated in the context of aligning two large, real-life thesauri: the UNs AGROVOC thesaurus (40 K terms) and the United States National Agricultural Library thesaurus NALT (65 K terms) [24]. The matching process performed with both strategies resulted in several thousand mappings, using several hundred online ontologies, with an average precision of 70%.


16.2.1.2 Swoogle Ontology Dictionary

Swoogle Ontology Dictionary is an add-on application on top of Swoogle. It collects all Semantic Web terms from the harvested Semantic Web documents and builds a global

view of the Semantic Web vocabulary. It has two potential contributions to the Semantic Web community:

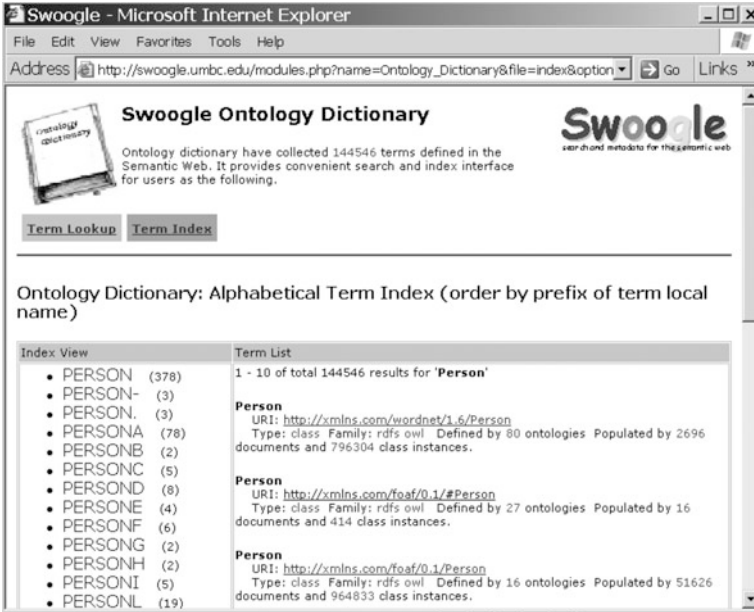
- It builds a comprehensive view of the Semantic Web vocabulary and breaks the (unnecessary) physical boundary imposed by Semantic Web ontologies. There are two well-known drawbacks of using ontology documents to group Semantic Web terms: (1) Semantic Web terms defined in one Semantic Web ontology may be instantiated in quite different frequencies, for example, *owl:versionInfo* is far less instantiated than *owl:Class* in the Semantic Web; and (2) Semantic Web terms from multiple ontologies are usually used together to modify one class-instance, for example, *rdfs:seeAlso* and *dc:title* have been frequently used together to modify the class-instances of *foaf:Person*.
- Beside the Semantic Web terms defined or referenced in Semantic Web ontologies, it also collects the Semantic Web terms which have been instantiated as classes or properties but have not been defined by any existing Semantic Web ontology. For example, the property <http://webns.net/mvcb/generatorAgent> has been widely used, and interested users may want to reuse this term even though no existing Semantic Web ontology has defined it.

Currently, Swoogle ontology dictionary provides two user interfaces for locating Semantic Web terms.

- *Term Search* is essentially a web interface based on the Swoogle term search API, which allows users to search SWTs by URI, namespace, local name, literal definitional description, and semantic definition.
- *Alphabetical Term Index*, as shown in  Fig. 16.11, organizes all Semantic Web terms by prefix alphabetically. It has two views: the *prefix view* (left panel) and the *matched-term-list view* (right panel). In the prefix view, each prefix is followed by the number of terms using that prefix (using case-insensitive string matching here). In the matched-term-list view, all terms matching the current prefix are listed.

16.2.1.3 Sig.Ma

Sig.Ma [25] (<http://sig.ma>) is a service built on top of the Sindice [26] Semantic Web search engine. Sindice indexes very large quantities of information from the Web, especially coming from the linked data community. Sig.Ma relies on Sindice to provide an aggregated view on the available semantic data for a given entity or resource. Starting from a simple keyword query supposed to describe the entity to look up; Sig.Ma displays the properties of the corresponding entities present in a large variety of linked data sources, as well as the correspondences between each piece of data and the sources where it originated. For example, using the name of a person as a starting point, Sig.Ma can show the location, photos, workplace, contact details, and birthday of this person, each piece of information potentially coming from a different source. Of course, noise could easily



■ Fig. 16.11

The alphabetical term index interface

appear in the results, due to the potential ambiguity of the initial query. Sig.Ma allows the user to customize the view by refining the list of sources, removing the ones which do not match the initial intent of the query.

A point worth noticing is that Sig.Ma is not only an application itself, but also provides a base for other applications. Each view, even customized, is associated with a Web address (a URI). An API and a widget are also available that give access to the functionalities of Sig.Ma to other applications.

16.2.1.4 The Watson Plug-In for Knowledge Reuse

Ontology reuse is a complex process involving activities such as searching for relevant ontologies for reuse, assessing the quality of the knowledge to reuse, selecting parts of it and, finally, integrating it in the current ontology project. As the Semantic Web provides more and more ontologies to reuse, there is an increasing need for tools supporting these activities.

The Watson plug-in (http://watson.kmi.open.ac.uk/editor_plugins.html) (see Fig. 16.12) aims to facilitate knowledge reuse by integrating the search capabilities of Watson within the environment of an ontology editor (the NeOn Toolkit, <http://neon-toolkit.org>). The resulting infrastructure allows the user to perform all the steps necessary for the large-scale reuse of online knowledge within the same environment where this knowledge is processed and engineered.

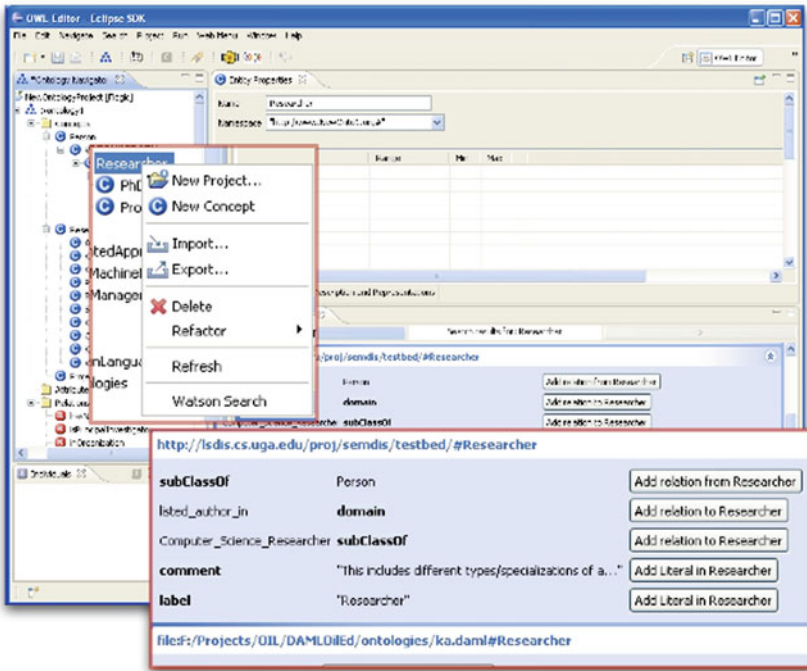


Fig. 16.12

The Watson Plug-in for ontology editors

In practice, the Watson plug-in allows the ontology developer to find, in existing online ontologies, descriptions of the entities present in the currently edited ontology (i.e., the *base* ontology), to inspect these descriptions (the statements attached to the entities) and to integrate these statements into the base ontology. For example, when extending the base ontology with statements about the class *Researcher*, the Watson plug-in identifies, through Watson, existing ontologies that contain relevant statements such as:

- *Researcher* is a subclass of *AcademicStaff*
- *PhDStudent* is a subclass of *Researcher*
- *Researcher* is the domain of the property *isAuthorOf*

These statements can be used to extend the edited ontology, integrating them to ensure, for example, that the class *Researcher* becomes a subclass of a newly integrated class *AcademicStaff*.

16.2.1.5 Swoogle-Based Triple Shop

Triple Shop [27] was developed to better assist users to utilize the search results of Swoogle. It worked as follows: Swoogle would present query results (URIs) to the user,

and then the user could check URIs to be added to his or her shopping cart. Eventually, a user could check out, have all URIs loaded into a triple store and be presented with an interface for issuing SPARQL queries. This utility proved to be an extremely useful tool in integrating scientific data. Below are some key features:

- **Finding datasets.** A dataset finder is a service that implements the Swoogle-assisted data access process by facilitating the completion of an incomplete SPARQL-ish query. Besides manually specifying the URIs of RDF resources, users can simply use English terms to refer to RDF resources in the WHERE clause of a SPARQL query. This service will search Swoogle for appropriate URIs to substitute the English terms in the query, and the user can then select one from the alternative resulting URIs. Users can also leave dataset specification empty, that is, without specifying the FROM clause. Again, the service will search Swoogle to suggest relevant SWDs to answer the query. It is notable that the search for SWDs and SWTs can be refined in a number of ways. Constraints can be placed on the domain of a URI, and on the namespaces that it uses.
- **Inference.** After constructing a dataset, the user can specify a level of reasoning to be performed in executing the query. Choices range from no reasoning, through RDFS, to OWL.
- **Dataset persistence and reuse.** A user can save a dataset on the Triple Shop server, tag a dataset, search for existing tagged datasets, and add tags to existing datasets. Each dataset can be stored as a list of URLs of SWDs, or be materialized into a merged RDF graph in triple store.

Triple Shop has been used in ELVIS (the Ecosystem Location Visualization and Information System), which is a suite of tools for constructing food webs for a given location. ELVIS is motivated by the belief that food web structure plays a role in the success or failure of potential species invasions. Because very few ecosystems have been the subject of empirical food web studies, response teams are typically unable to get quick answers to questions like what are likely prey and predator species of the invader in the new environment? The ELVIS tools seek to fill this gap. ELVIS functionality is exposed as a collection of Web Services, and all input and output data are expressed in OWL, thereby enabling its integration with other Semantic Web resources.

Bioinformatic data in ELVIS are encoded in RDF and cover the following categories: (1) species distribution data compiled by the California node of the National Biological Information Infrastructure; (2) trophic data compiled from over 250 datasets; (3) the complete contents of Animal Diversity Web (ADW), a popular online encyclopedia [48]; and (4) a collection of lists designating species as being invasive in particular regions.

With the available datasets in ELVIS, researchers can verify their hypotheses on the complex relations in a food web. The complex relations can be mapped to a query on the RDF data in ELVIS. As the researchers may not necessarily know URIs for all the terms or know which datasets are relevant, Triple Shop can assist completing a query with the help of Swoogle, gathers/integrates all triples that might be relevant to the query, and will do forward-chaining inference to generate all implied triples when appropriate. This

process may take anywhere from seconds to hours. When it is complete, the researchers can see query results, and share the resulting integrated dataset with colleagues in a persistent manner.

16.2.1.6 Evolva: Ontology Evolution Using Background Knowledge

Ontologies form the backbone of Semantic Web–enabled information systems. Today’s organizations generate huge amounts of information daily, thus ontologies need to be kept up to date in order to reflect the changes that affect the life cycle of such systems (e.g., changes in the underlying datasets, a need for new functionalities, etc.). This task, described as the “timely adaptation of an ontology to the arisen changes and the consistent management of these changes,” is called *ontology evolution* [28]. While it seems necessary to apply such a process consistently for most of the ontology-based systems, it is often a time-consuming and knowledge-intensive task, as it requires a knowledge engineer to identify the need for change, perform appropriate changes on the base ontology, and manage its various versions.

Evolva (an overview of Evolva can be found in [29, 30]) is an ontology evolution system starting from external data sources (text documents, folksonomies, databases, etc.) that form the most common means of storing data. First, a set of terms are extracted from these sources as potentially relevant concepts/instances to add to the ontology, using common information extraction methods. Evolva then makes use of Watson (through the intermediary of Scarlet) to find external sources of background knowledge to establish relations between these terms and the knowledge already present in the ontology, providing in this way the means to integrate these new terms in the ontology. For this purpose, a relation discovery process was devised, that combines various background knowledge sources with the goal of optimizing time-performance and precision.

16.2.1.7 Wahoo/Gowgle: Query Expansion

Wahoo and Gowgle (<http://watson.kmi.open.ac.uk/wahoo> and <http://watson.kmi.open.ac.uk/gowgle>) are two demonstrators, showing how Watson can be used for a simple application to perform query expansion in a classical Web search engine. For example, when given the keyword *developer*, such a tool could find out that in an ontology, there is a subclass *programmer* of *developer* and could therefore suggest this term as a way to specify the query to the Web search engine. Without Watson, this would require one to integrate one or several ontologies about the domain of the queries and an infrastructure to store them, explore them, and query them. However, if the considered search engine is a general Web search engine, such as Google or Yahoo!, the domain of the queries cannot be predicted: the appropriate ontology can only be selected at runtime, depending on the query that is given. In addition, this application would require a heavy infrastructure to be

able to handle large ontologies and to query them efficiently. Gowgle and Wahoo rely on Semantic Web ontologies explored using Watson instead.

The overall architecture of these applications is made of a Javascript/HTML page for entering the query and displaying the results, which communicates using the principles of AJAX with the Watson server. In the case of Gowgle, Google is used as the Web search engine and the Watson SOAP Web Services are employed for ontology exploration (http://watson.kmi.open.ac.uk/WS_and_API.html). In the case of Wahoo, Yahoo!, and the Watson REST API (http://watson.kmi.open.ac.uk/REST_API.html) are used.

Both applications use Watson to exploit online ontologies in order to suggest terms related to the query, that is, if the query contains the word *developer* (1) to find ontologies somewhere talking about the concept of developer, (2) to find in these ontologies which entities correspond to *developer*, and (3) to inspect the relations of these entities to find related terms.

16.2.1.8 SWAML

SWAML (<http://swaml.berlios.de/>), the Semantic Web Archive of Mailing Lists Project [31], is building a series of tools to enable the semantic publication and browsing of e-mail collections. It is able to extract e-mails from a mailbox and create a representation of these e-mails using mainly the SIOC ontology (<http://sioc-project.org/>). However, the information contained in the mailbox alone is not enough to organize its content. People information, for example, is present in many different sources on the Semantic Web, based on the FOAF vocabulary (<http://www.foaf-project.org/>). SWAML, therefore, uses Sindice to collect semantic data related to people, based on their e-mail addresses. One of the advantages of Sindice in this case is its ability to draw inferences on inverse functional properties (IFPs). Indeed, in FOAF, the relation connecting a person to his or her e-mail address is declared as an IFP, meaning that an e-mail address is associated to only one person. Therefore, whenever several resources appear to be connected to the same e-mail address, Sindice can infer that these resources refer to the same person.

16.2.1.9 PowerAqua: Question Answering

To some extent, PowerAqua (<http://poweraqua.open.ac.uk/>) can be seen as a straightforward human interface to any semantic document indexed by Watson. Using PowerAqua, a user can simply ask a question, like “Who are the members of the rock band Nirvana?” and obtain an answer, in this case in the form of a list of musicians (Kurt Cobain, Dave Grohl, Krist Novoselic, and other former members of the group). The main strength of PowerAqua resides in the fact that this answer is derived dynamically from the relevant datasets available on the Semantic Web.

Without going into too many details, PowerAqua first uses a Gate-based [32] linguistic component to transform a question into a set of possible “query triples,” such as <person/organization, members, rock band Nirvana>. The next step consists then in locating,

thanks to Watson, online semantic documents describing entities that correspond to the terms of the query triples, locating, for example, an individual called *Nirvana* in a dataset about music. During this step, WordNet (<http://wordnet.pinceton.edu>) is used to augment the terms in the query triples with possible synonyms. Once a collection, usually rather large, of potential candidate ontologies is found, PowerAqua then employs a variety of heuristics and a powerful matching algorithm, PowerMap [33], to try and find answers from the collection of candidate ontologies. In the example, the query triple shown above can be successfully matched to the schema $\langle \text{Nirvana, has_members, ?x:Musician} \rangle$, which has been found in a music ontology on the Semantic Web. In more complex examples, an answer may require integrating a number of statements. For instance, to answer a query such as “Which Russian rivers flow to the Black Sea,” PowerAqua may need to find information about Russian rivers, information about rivers which flow to the Black Sea and then combine the two. In general, several sources of information, coming from various places on the Web, may provide overlapping or complementary answers. These are therefore ranked and merged according to PowerAqua’s confidence in their contribution to the final answer.

16.2.1.10 PowerMagpie: Semantic Browsing

PowerMagpie (<http://powermagpie.open.ac.uk>) is a Semantic Web browser that makes use of openly available semantic data through Watson to support the interpretation process of the content of arbitrary Web pages. Unlike its predecessor, Magpie, which relied on a single ontology selected at design time, PowerMagpie automatically, that is, at runtime, identifies and uses relevant knowledge provided by multiple online ontologies. From a user perspective, PowerMagpie is an extension of a classical Web browser and takes the form of a vertical widget displayed on top of the currently browsed Web page. This widget provides several functionalities that allow the exploration of the semantic information relevant to the current Web page. In particular, it summarizes conceptual entities relevant to the Web page. Each of the entities can then be shown in the text, where the user may initialize different ways of exploring the information space around a particular entity. In addition, the semantic information discovered by PowerMagpie, which relates the text to online semantic resources, is “injected” into the Web page as embedded annotations in RDFa. These annotations can then be stored into a local knowledge base and act as an intermediary for the interaction of different semantic-based systems.

16.2.1.11 FLOR: Folksonomy Ontology Enrichment

Folksonomies, social tagging systems such as Flickr and Delicious, are at the forefront of the Web2.0 phenomenon as they allow users to tag, organize, and share a variety of information artifacts. The lightweight structures that emerge from these tag spaces only weakly support content retrieval and integration applications since they are agnostic to

the explicit semantics underlying the tags and the relations among them. For example, a search for *mammal* ignores all resources that are not tagged with this exact word, even if they are tagged with specific mammal names such as *lion*, *cow*, and *cat*. The objective of FLOR [34] is to attach formal semantics to tags, derived from online ontologies and make the relations between tags explicit (e.g., that *mammal* is a superclass of *lion*). The enrichment algorithm that has been experimentally investigated builds on Watson: given a set of tags, the prototype identifies the ontological entities (classes, properties, and individuals) that define the tags in their respective contexts. Additionally, it aims to identify formal relations between the tags (subsumption, disjointness, and generic relations) utilizing Scarlet.

The experiments [21] have led to further insights into the nature of ontologies on the Semantic Web, from which two key ones are highlighted here. First, it was found that online ontologies have a poor coverage of a variety of tag types denoting novel scientific terminology, multilingual terms, and domain-specific jargon. Secondly, it was observed that online ontologies can reflect different views and when used in combination can lead to inconsistencies in the derived structures.

16.2.1.12 The Watson Synonym Service

The Watson Synonym Service (<http://watson.kmi.open.ac.uk/API/term/synonyms>) is a simple service that creates a base of term clusters, where the terms of a cluster are supposed to be associated to the same sense. It makes use of the information collected by Watson in the form of ontologies to derive these clusters.

The basic algorithm to create term clusters is quite straightforward. Entities in Semantic Web ontologies all possess one and only one identifier (in a given namespace, e.g., *Person* is considered to be the identifier of <http://www.example.org/onto#Person>). They can also be associated to one or several labels, through the `rdf:label` property. Hence, the algorithm simply assumes that a term t_1 is a synonym of another term t_2 if t_1 and t_2 are used either as label or identifier of the same entity. The role of the synonym discovery offline algorithm is then simply to iterate through all the entities in Watson's ontologies to create clusters of terms that are used together in the identifiers or labels of entities.

Of course, the quality of the results obtained with this method is not as good as the one obtained with the complex and costly approaches that are employed to build systems such as WordNet (<http://wordnet.pinceton.edu>). However, the advantage of this algorithm is that its quality improves together with the growth of the Semantic Web, without requiring any additional effort for collecting the data. A high number of good synonyms are found, like in the cluster {*ending*, *death*, *termination*, *destruction*}. In addition, this method does not only find synonyms in one language, but can provide the equivalent terms in various languages, providing that multilingual ontologies exist and cover these terms. It could be argued that these are not actually synonyms (but translations) and one of the possible extensions for this tool is to make use of the language information in the ontologies to distinguish these cases.

16.2.2 Semantic Web Search Engines as Research Platforms

Semantic Web search engines are tools and infrastructure components that automatically collect, analyze, and index ontologies and semantic data available online. Besides enabling the exploitation of the Semantic Web, they can be seen as a research platform supporting the exploration of the Semantic Web to better understand its characteristics. Indeed, most of the existing systems provide statistics for the documents and Semantic Web entities they have collected (see, e.g., the statistics page of the Falcons system, <http://iws.seu.edu.cn/services/falcons/statistics.jsp>), but beyond basic statistics, researchers involved in the development of Semantic Web search engines were able to realize global studies of the Semantic Web landscape, using the large collections of ontologies and semantic data available through these systems.

16.2.2.1 Swoogle-Based Semantic Web Statistics

Based on the Semantic Web dataset collected by Swoogle, measures of some statistical properties of Semantic Web data were presented in [35]. This paper should be considered for precise results available at the time of its publication; however, the focus here is on demonstrating how Swoogle can be used to compute these measures, as the actual values would need to be updated to reflect the current status of the Semantic Web.

One interesting question is the size of the Semantic Web on the Web. However, this number is hard to obtain because (1) Semantic Web documents are sparsely distributed on the Web and (2) validating whether a Web document is a Semantic Web document requires nontrivial computation. Brute-force sampling, that is, measuring the size of the Web (e.g., testing 80 ports for a huge list of IP addresses) [36], is not suitable due to their unacceptable low efficiency. Analysis on the overlap of meta-search results of conventional Web search engines [17, 37] is suitable mainly because SWDs are less favored by these engines, and some even provide limited support on searching SWDs. For example, even though both support filetype search, only Google search but not MSN search supports searching for the filetype “rdf” and “owl.” A Google-based meta-search is adopted for estimating SWDs based on the observation that 99% of SWDs have declared *RDF namespace*, whose URL is <http://www.w3.org/1999/02/22-rdf-syntax-ns#>, as non-markup which should be indexed by conventional search engines.

Another interesting measure is the deployment status of the Semantic Web on the Web with respect to the Web and the RDF graph world. In particular, a series of quantitative metrics and in-depth analysis bring a global picture of the SWDs and SWTs in the Semantic Web. (Invariant) Power distribution has been observed in many cases, such as the distribution of SWDs per website and the definition quality of SWT. It was also noticed that the bias introduced by the dynamic SWDs could block the diversity of the Semantic Web and should be controlled. A good number of metrics have been proposed for measuring the statistical distribution of SWDs and SWTs. SWDs are the atomic containers for transferring Semantic Web data and the interfaces between the Web and the RDF graph world.

- Source of SWD. In order to measure how Semantic Web data are distributed on the Web, SWDs are grouped by their source websites. SWDs can further be grouped by the top-level domain extracted from the URLs of the website hosting the SWDs.
- Size of SWD. The size of a SWD indicates the volume of Semantic Web data in the SWD, which is usually measured by the number of triples in the RDF graph parsed from the SWD.
- Age of SWD. SWDs could be uploaded, modified, and removed on the Web. The age of an SWD is measured by the last modified time (attached in the header of HTTP response) of its latest version.
- Size change of SWD. In order to track the size change of SWDs, snapshots of each SWD are maintained once a new version has been detected.
- Definition quality of SWD. In order to evaluate the portion of the definition in an SWD, the *ontology ratio* (OntoRatio) is calculated at class-instance level and triple level. High *OntoRatio* implies a preference for adding term definition rather than populating existing terms; hence, *OntoRatio* can be used to quantify the degree of a Semantic Web document being a “real” ontology.

SWTs are also evaluated using collected data.

- Overall Meta-Usage of SWT. Analyzes the usage of SWTs in SWDs based on the combination of the six types of meta-usage identified by the WOB ontology, namely, *hasClassDefinitionIn*, *hasPropertyDefinitionIn*, *hasClassInstanceIn*, *hasPropertyInstanceIn*, *hasClassReferenceIn*, and *hasPropertyReferenceIn*.
- Definition quality of SWT. The definition of an SWT depends on its residential RDF graph that is serialized by an SWD. Again, the number of definitional triples of the SWT is counted to estimate the quality of its definition within an SWD. Usually, important classes and properties have more definitional triples.
- A common question posed by Semantic Web knowledge consumers is what kind of Semantic Web data are available. The answer to this question is given by measuring the instance space of the Semantic Web, that is, how SWTs are populated in SWDs as classes and properties, for example, the number of SWTs being populated as class (or property) by at least m instances,

The navigational paths in the Semantic Web are still in small amount and not enough for effective Semantic Web surfing. In the category, Navigation Quality using statistics of several important types of paths was investigated: (1) paths based on explicit import semantics, (2) paths based on inexplicit namespace reference, and (3) paths based on Link Indicators, such as the value of *rdfs:seeAlso* in FOAF.

16.2.2.2 Characterizing Knowledge on the Web with Watson

To give an account of the way semantic technologies are used to publish knowledge on the Web, of the characteristics of the published knowledge, and of the networked aspects of

the Semantic Web, an analysis of a sample of 25,500 semantic documents collected by Watson was realized (see [38] for the details).

This analysis looked in particular into the use of Semantic Web languages and of their primitives. Watson implements a simple, but restrictive language detection mechanism. It is restrictive in the sense that it considers a document to employ a particular language only if this document actually *instantiates* an entity of the language vocabulary (any kind of description for RDF, a class for RDF-S, and a class or a property for OWL and DAML + OIL). A simple conclusion that can be drawn from this analysis is that, while the majority of the considered documents are exclusively considering factual data in RDF, amongst the ontology representation languages (RDF-S, OWL and DAML + OIL), OWL has clearly been adopted in majority.

The initial version of OWL was divided into three sub-languages, OWL Lite, OWL DL, and OWL Full, that represent different (increasing) levels of complexity (in the current version, OWL 2, these sub-languages have been replaced by *profiles*, see <http://www.w3.org/TR/owl2-profiles/> see also ► [KR and Reasoning on the Semantic Web: OWL](#)). Another way to measure the expressivity (and so the complexity) of the language used is to consider the underlying description logic. Description logics are named according to the primitives they contain. For example, the DL of OWL Lite is $\mathcal{ALCCR}_+ \mathcal{HIF}(D)$, meaning, for example, that it allows the description of inverse relations (\mathcal{I}) and of limited cardinality restrictions (\mathcal{F}). One noticeable fact that can be derived from analyzing both the OWL Species and the description logic used in ontologies is that, while a large majority of the ontologies in the set were in OWL Full (the most complex variant of OWL, which is undecidable), most of them were in reality very simple, only using a small subset of the primitives offered by the language (95% of the ontologies were based on the $\mathcal{ALH}(D)$ description logic). This is consistent with conclusions obtained in [39].

Looking at the size and structure of Semantic Web documents also highlighted that a large majority of them were very simple. Indeed, a simple measure of density for RDF entities is used (measuring relations they share with other entities) and discovered that the employed collection of online semantic documents was made of a very large number of very small and very shallow structures, and of a very small number of very large and complex ontologies.

Another interesting element to consider is the duplication of URIs. Indeed, in theory, if two semantic documents are identified by the same URI, they are supposed to contribute to the same ontology, that is, the entities declared in these documents are intended to belong to the same conceptual model. However, even if this situation appears rarely (only 60 URIs of documents are “nonunique” in the considered set), in most cases, semantic documents that are identified by the same URI are not intended to be considered together. Different situations can be distinguished that lead to this problem:

Default URI of the ontology editor: <http://a.com/ontology> is the URI of 20 documents that do not seem to have any relation with each other, and that are certainly not meant to be considered together in the same ontology. The reason for this URI to be so popular is that it was the default namespace attributed to ontologies edited using the Protégé

editor (<http://protege.stanford.edu/>) at the time. This problem has been reduced now by the fact that Protégé forces its users to change the URI of their ontologies.

Mistaken use of well-known namespaces: The second most commonly shared URI in the Watson repository is <http://www.w3.org/2002/07/owl>, which is the URI of the OWL schema. The namespaces of RDF, RDF Schema, and of other well-known vocabularies are also often duplicated. Using these namespaces as URIs for ontologies is (in most cases) a mistake that could be avoided by checking, prior to giving an identifier to an ontology, if this identifier has already been used in another ontology.

Different versions of the same ontology: A third common reason for which different semantic documents share the same URI is in situations where an ontology evolves to a new version, keeping the same URI (e.g., <http://lsdis.cs.uga.edu/proj/semdis/testbed/>). As it is the same ontology, it seems natural to keep the same URI, but in practice, this can cause problems in these cases where different versions coexist and are used at the same time. This leads to a need for recommendations of good practices on the identification of ontologies, that would take into account the evolution of the ontologies, while keeping different versions clearly separated.

Related to this last point, an initial experiment [40] recently investigated the use of information encoded in the URIs of the ontologies to encode versioning data, which can be extracted to trace the different versions of ontologies. It appears that many different, more or less popular conventions are used to encode such version data, from the use of version numbers (e.g., `v1.2`, `rev = 3.6`) to the use of time-stamps and dates (using two or three numbers, in big endian or little endian orders). Through recognizing these patterns in URIs, many “chains” of ontology versions can be detected with varying levels of accuracy, providing an insight on how ontologies evolve on the Web.

Watson provides an efficient platform, allowing researchers to obtain an overview of the Semantic Web, to apprehend its content and development, and to analyze the way knowledge is published online. Many other elements have been, and could be analyzed concerning the Semantic Web, including the (explicit and implicit) relationships existing between documents, the coverage in terms of domains and topics, etc. [41]. The next section briefly summarizes recent work on using Watson to measure agreements and disagreements in ontologies.

16.2.2.3 Measuring Ontology Agreement and Disagreement in Watson

Ontologies are knowledge artifacts representing particular models of some particular domains. They are built within the communities that rely on them, meaning that they represent consensual representations inside these communities. However, when considering the set of ontologies distributed on the Web, many different ontologies can cover the same domain, while being built by and for different communities. Knowing which ontologies agree or disagree with others or how much a particular statement is generally agreed with in online ontologies can be very useful in many scenarios.

One way to detect whether there is a disagreement between two ontologies is to rely on the presence of logical contradictions. The two ontologies can be merged, based on mappings between their entities, and the resulting model be checked for inconsistencies and incoherences. While this approach would certainly detect some forms of disagreement, it only checks whether the ontologies disagree or not. It does not provide any granular notion of disagreement and, if no contradictions are detected, it does not necessarily mean that the ontologies agree. Indeed, while two ontologies about two completely different, nonoverlapping domains would certainly not disagree, they do not agree either. More importantly, logical contradictions are not the only way for two ontologies to disagree. Indeed, there could also be conceptual mismatches, like in the case where one ontology declares that “Lion is a subclass of Species” and the other one indicates that “Lion is an instance of Species.” Even at content level, logical contradictions would not detect some form of disagreements. Indeed, the two statements “Human is a subclass of Animal” and “Animal is a subclass of Human” do not generate any incoherence. However, they disagree in the sense that, if put together, they generate results that were not expected from any of the two ontologies.

For these reasons, [42] defines two basic measures for assessing agreement and disagreement of an ontology O with a statement $s = \langle \text{subject}, \text{relation}, \text{object} \rangle$:

$$\begin{aligned} \text{agreement}(O, s) &\rightarrow [0..1] \\ \text{disagreement}(O, s) &\rightarrow [0..1] \end{aligned}$$

Two distinct measures are used for agreement and disagreement so that an ontology can, at the same time and to certain extents, agree and disagree with a statement. These two measures have to be interpreted together to indicate the particular belief expressed by the ontology O regarding the statement s . For example, if $\text{agreement}(O, s) = 1$ and $\text{disagreement}(O, s) = 0$, it means that O fully agrees with s and conversely if $\text{agreement}(O, s) = 0$ and $\text{disagreement}(O, s) = 1$, it fully disagrees with s . Now, agreement and disagreement can vary between 0 and 1, meaning that O can only partially agree or disagree with s and sometimes both, when $\text{agreement}(O, s) > 0$ and $\text{disagreement}(O, s) > 0$. Finally, another case is when $\text{agreement}(O, s) = 0$ and $\text{disagreement}(O, s) = 0$. This basically means that O neither agrees nor disagrees with s , for the reason that it does not express any belief regarding the relation encoded by s .

The actual values returned for both measures, when different from 0 and 1, are not very important. They correspond to different levels of disagreement/agreement and only an order between predefined levels is needed to interpret them. The values used and the ways to compute them are given in [42].

Considering that ontologies are made of statements, extending the measures above to compute agreement and disagreement between two ontologies is relatively straightforward, using the mean of each measure for each statement of an ontology against the other ontology, in both directions and making this a normalized measure. However, while relatively simple, the two measures of agreement and disagreement between ontologies provide an interesting way to obtain an overview of a set of ontologies. Indeed, an experiment looked at the 21 ontologies returned by Watson when querying for semantic documents containing a class with the term *SeaFood* in its ID or label, and computed the agreement and disagreement

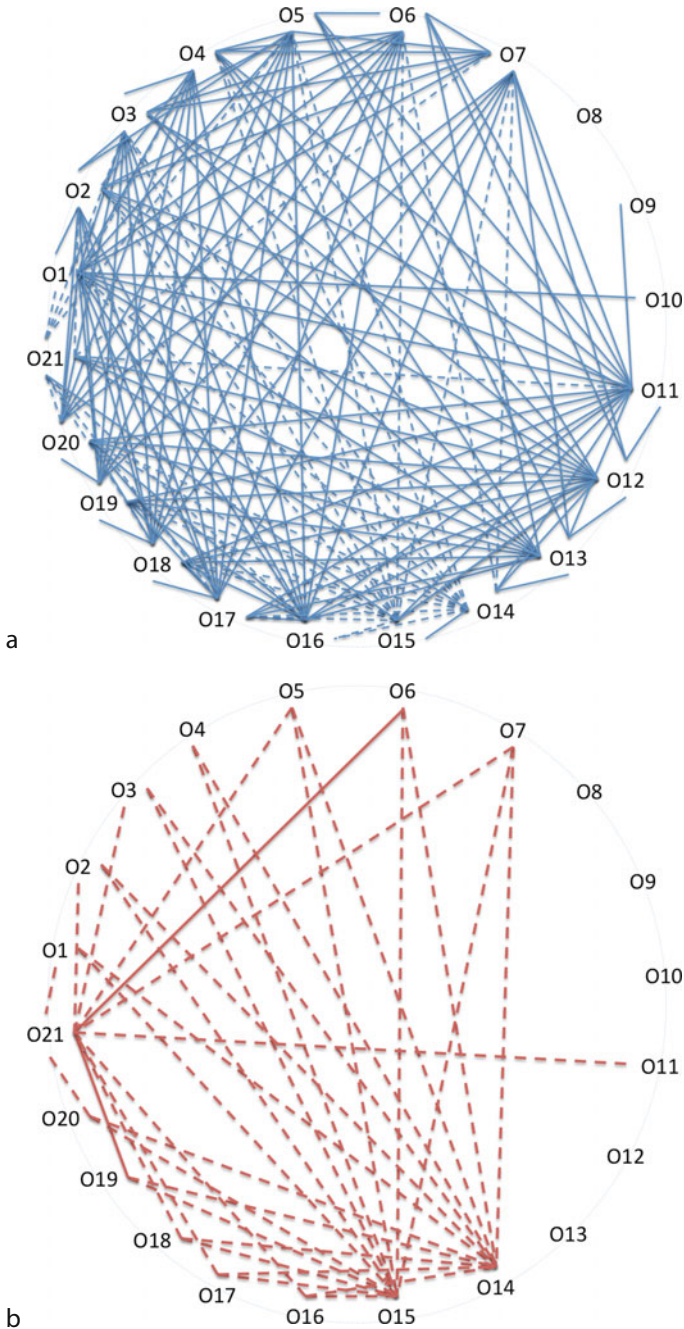


Fig. 16.13

Agreement (*top*) and disagreement (*bottom*) relations among the 21 test ontologies. *Plain lines* represent full disagreement/agreement (measures' values = 1). *Dashed lines* represent partial disagreement/agreement (measures' values greater than 0)

measures for all pairs of ontologies in this set. The results are shown in [Fig. 16.13](#) where ontologies are numbered according to their rank in Watson (valid on the 20/09/2009).

Analyzing these diagrams, it appears that there is a certain level of “coherence” in the results. In particular, homogeneous clusters can be built from the agreement and disagreement values: the ontologies O1, O2, O3, O4, O5, O6, O7, O11, O12, O13, O16, O17, O18, O19, and O20 all fully agree with each other and, at the same time, partially agree and disagree with O14 and O15. O14 and O15 also form a cluster since they agree with each other, and consistently disagree with the same set of ontologies (the reason being that O14 and O15 are the ontologies considering that SeaFood is a subclass of Meat, but agree on all the other related statements). O21 is also particular, since it disagrees with most of the ontologies of the first cluster, sometimes fully. Indeed, it also considers SeaFood to be a subclass of Meat, and additionally disagrees on several other statements with some of the other ontologies (e.g., it considers that tuna is a subclass of fish while several other ontologies consider tuna as an instance of fish). O8, O9, and O10 are particular since there is only a very small overlap between them and the other ontologies. For example, O9 only agrees with O11 that Vegan is a subclass of Vegetarian.

Another interesting piece of information that can be derived from the measures defined and from exploiting the collection of ontologies in Watson is the level to which particular statements are agreed with, that is, the level of consensus on a statement. Conversely, a related item of information concerns the level of controversy on the statement, that is, whether there is a clear-cut between agreement and disagreement. Here, a normalized mean was also used to measure the global agreement and disagreement of a statement st in a set of ontologies R (see details in [42]). From these two measures, consensus is defined as having a high level of certainty on whether ontologies in R agree or disagree with st . There is a high level of (positive consensus) if the overall agreement about this statement is high and the overall disagreement is low. Thus, the measure of consensus is computed in a set of ontologies R upon a statement st as follows:

$$\text{consensus}(st, R) = \text{agreement}(st, R) - \text{disagreement}(st, R)$$

The notion of controversy is then considered to be the inverse from the one of consensus: there is a high level of controversy on a given statement when there is no clear-cut between agreement and disagreement, that is, there is a low level of consensus. Therefore, the measure of controversy in a set of ontologies R upon a statement st can simply be computed in the following way:

$$\text{controversy}(st, R) = 1 - |\text{consensus}(st, R)|$$

To illustrate these measures, nine statements concerning the class *SeaFood* in Watson are considered. The results are summarized in [Table 16.1](#).

As can be seen from these results, the first four statements are fully agreed with by ontologies in Watson, meaning that all the ontologies containing both entities of each statement express exactly the same relation as the one of the statement. The three next statements also have a very high level of agreement, and a very low level of disagreement. This is mainly due to a few ontologies containing the right entities, but not necessarily

■ **Table 16.1**

Consensus and controversy on statements concerning the *SeaFood* class in Watson

Statement	Consensus	Controversy
< <i>SeaFood</i> , <i>disjointWith</i> , <i>Dessert</i> >	1.0	0.0
< <i>Fowl</i> , <i>disjointWith</i> , <i>SeaFood</i> >	1.0	0.0
< <i>Pasta</i> , <i>disjointWith</i> , <i>SeaFood</i> >	1.0	0.0
< <i>SeaFood</i> , <i>subClassOf</i> , <i>EdibleThing</i> >	1.0	0.0
< <i>ShellFish</i> , <i>subClassOf</i> , <i>SeaFood</i> >	0.89	0.109
< <i>Fish</i> , <i>subClassOf</i> , <i>SeaFood</i> >	0.875	0.125
< <i>SeaFood</i> , <i>disjointWith</i> , <i>Fruit</i> >	0.75	0.25
< <i>Meat</i> , <i>disjointWith</i> , <i>SeaFood</i> >	0.53	0.46
< <i>SeaFood</i> , <i>subClassOf</i> , <i>Meat</i> >	-0.719	0.281

describing any relation between them. Hence, there is a high level of consensus on these statements. Finally, the last two statements are the ones for which there is the highest level of controversy. The last one is by far the most disagreed with (which correlates with the high level of agreement of the other one contradicting it).

Another interesting example is the one of the statement < *river*, *subClassOf*, *sea* >, which gives a high level of disagreement (0.766). The disagreement is not 1 in that case, because only very few ontologies express explicitly contradicting relations. However, in this case, the level of agreement is 0: There is no ontology to actually agree with this statement.

16.3 Related Resources

The previous sections give a detailed account of existing uses and applications of Semantic Web search engines, focusing in particular on two of the most prominent systems which are currently active. Due to the increase in the number of semantic documents made available online, and so to the need for search functionalities, a number of other systems have emerged recently from academic research (the list is deliberately restricted to systems that provide at least a freely accessible Web user interface for searching or querying semantic data):

Sindice (<http://sindice.com/>) is a *Semantic Web index* or *entity look-up service* that focuses on scaling to very large quantities of data. It provides keyword and URI-based search, structured query, and relies on some simple reasoning mechanisms for inverse-functional properties [26].

Falcons (<http://iws.seu.edu.cn/services/falcons/>) is a keyword-based semantic entity search engine. It provides a sophisticated Web interface that allows one to restrict the search according to recommended concepts or vocabularies [43].

SWSE (<http://swse.deri.org/>) is also a keyword-based entity search engine, but one that focuses on providing semantic information about the resulting entities rather than only links to the corresponding data sources [44]. Its collection is automatically

gathered by crawlers. SWSE also provides a SPARQL endpoint enabling structured query on the entire collection.

Semantic Web Search (<http://www.semanticwebsearch.com/>) is also a semantic entity search engine based on keywords, but that allows one to restrict the search to particular types of entities (e.g., DOAP Projects) and provides structured queries.

OntoSelect (<http://olp.dfki.de/ontoselect/>) provides a browsable collection of ontologies that can be searched by looking at keywords in the title of the ontology or by providing a topic [45].

OntoSearch2 (<http://www.ontosearch.org/>) is a Semantic Web Search engine that allows for keyword search, formal queries, and fuzzy queries on a collection of manually submitted OWL ontologies. It relies on scalable reasoning capabilities based on a reduction of OWL ontologies in DL-Lite ontologies [46].

Sqore (<http://ict.shinawatra.ac.th:8080/sqore>) is a prototype search engine that allows for structured queries in the form of OWL descriptions [47]. Desired properties of entities to be found in ontologies are described as OWL entities and the engine searches for similar descriptions in its collection.

Finally, it is worth noticing that the issue of collecting semantic data from the Web has recently reached a broader scope, with the appearance of features within mainstream Web search engines exploiting structured data to improve the search experience and presentation. Indeed, Yahoo! SearchMonkey (<http://developer.yahoo.com/searchmonkey/>) crawls and indexes semantic information embedded in Web pages as RDFa (<http://www.w3.org/TR/xhtml-rdfa-primer/>) or microformats (<http://microformats.org/>), in order to provide enriched snippets describing the Web pages in the search results. Similarly, Google Rich Snippets (<http://googlewebmastercentral.blogspot.com/2009/05/introducing-rich-snippets.html>) makes use of collected semantic data using specific schemas in Web pages to add information to the presentation of results.

16.4 Conclusion and Future Directions

Semantic Web search engines are critical to the Semantic Web infrastructure. With the growth of Semantic Web data, applications and users, more and more research and development activities are being dedicated to building robust and scalable Semantic Web search engines. Most of the resulting systems are comparable in their structures and goals, but take different perspectives on the type of content they collect, on the task they support, and on the techniques they implement.

Developing such a system is a fascinating experience, touching on many different practical aspects of Semantic Web developments and including elements from other areas (information retrieval, interaction, databases, Web development, etc.), while integrating the tough constraint of reliability. But even more fascinating is the way Semantic Web search engines are used. They enable a new generation of applications that can benefit from a body of knowledge comparable to no other before. They allow users to explore this

knowledge in efficient ways. They form a platform for researchers to study the Semantic Web and understand its content, its structure, and its evolution.

While Semantic Web search engines have gone a long way since the very first version of Swoogle (in 2004!), many research issues still need to be explored. The dynamic aspect of the Semantic Web will certainly become an important problem in the next few years and Semantic Web search engines will be required to come up with new solutions to deliver only valid, up-to-date knowledge. The implicit relationships that relate semantic documents should also be better explored, providing ways to really exploit the network of ontologies which is available online, in a currently very shallow form. Also, while the quality of online information is still a major issue, facilitating various levels of user contributions, from writing new ontologies to linking datasets and reviewing semantic information, seems an interesting direction for the future.

16.5 Cross-References

- KR and Reasoning on the Semantic Web: OWL
- KR and Reasoning on the Semantic Web: Web-scale Reasoning
- Ontologies and the Semantic Web
- Querying the Semantic Web: SPARQL
- Semantic Annotation and Retrieval: RDF
- Semantic Annotation and Retrieval: Web of Data
- Semantic Annotation and Retrieval: Web of Hypertext – RDFa and Microformats
- Semantic Web Architecture
- Storing the Semantic Web: Repositories

References

1. Sowa, J.F.: Conceptual graphs summary. In: Nagle, T.E., Nagle, J.A., Gerholz, L.L., Eklund, P.W. (eds.) *Conceptual Structures: Current Research and Practice*, pp. 3–51. Ellis Horwood, New York (1992) ISBN:0-13-175878-0
2. Hobbs, J.R., Ferguson, G., Allen, J., Fikes, R., Hayes, P., McDermott, D., Niles, I.: Adam Pease, Austin Tate, Mabry Tyson, Richard Waldinger. A daml ontology of time. <http://www.cs.rochester.edu/~ferguson/daml/daml-time-nov2002.txt> (2002)
3. Chen, H., Perich, F., Finin, T., Joshi, A.: SOUPA: standard ontology for ubiquitous and pervasive applications. In: *Proceedings of the First International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQitous 2004)*, Boston (2004)
4. Hayes, P.: RDF semantics. <http://www.w3.org/TR/2004/REC-rdf-nt-20040210/> (2004)
5. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL web ontology language semantics and abstract syntax. <http://www.w3.org/TR/2004/REC-owl-semantics-20040210/> (2004)
6. Ayers, D., Völkel, M.: Cool URIs for the semantic web, W3C Interest Group Note. <http://www.w3.org/TR/cooluris/> (Sept 2010)
7. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF. <http://www.w3.org/TR/2006/WD-rdf-sparql-query-20060220/> (2006)
8. Bizer, C.: The emerging web of linked data. *IEEE Intell. Syst.* **24**, 87–92 (2009)
9. Bizer, C., Heath, T., Berners-Lee, T.: Linked data, the story so far. *Int. J. Semantic Web Inf. Syst.* **5**(3), 1–22 (2009)
10. Androutsopoulos, I., Ritchie, G.D., Thanisch, P.: Natural language interfaces to databases: an introduction. *Nat. Lang. Eng.* **1**(1), 29–81 (1995)

11. Harth, A., Umbrich, J., Decker, S.: Multi-crawler: a pipelined architecture for crawling and indexing semantic web data. In: Proceedings of the Fifth International Semantic Web Conference (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 258–271. Springer, Berlin (2006)
12. Ding, L., Pan, R., Finin, T., Joshi, A., Peng, Y., Kolari, P.: Finding and ranking knowledge on the semantic web. In: Proceedings of the Fourth International Semantic Web Conference (ISWC 2005), Galway. Lecture Notes in Computer Science, vol. 3729, pp. 156–170. Springer, Heidelberg (2005)
13. Alani, H., Brewster, C., Shadbolt, N.: Ranking ontologies with aktiverank. In: Proceedings of the Fifth International Semantic Web Conference (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 1–15. Springer, Berlin (2006)
14. d'Aquin, M., Euzenat, J., Le Duc, C., Lewen, H.: Sharing and reusing aligned ontologies with cupboard. In: Demo, Proceedings of the Fifth International Conference on Knowledge Capture (K-CAP 2009), Los Angeles (2009)
15. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V., and Sachs, J.: Swoogle: a search and metadata engine for the semantic web. In: Proceedings of the 13th ACM International Conference on Information and Knowledge Management (CIKM 2004), Washington, DC, pp. 652–659 (2004)
16. Sherman, C.: Metacrawlers and metasearch engines. <http://searchenginewatch.com/links/article.php/2156241> (last visited on March 2006) (2004)
17. Gulli, A., Signorini, A.: The indexable web is more than 11.5 billion pages. In: Proceedings of the 14th International World Wide Web Conference (WWW 2005) (poster paper), Chiba (2005)
18. Page, L., Brin, S., Motwani, R., Wino-grad, T.: The PageRank citation ranking: bringing order to the web. Technical report, Stanford Digital Library Technologies Project (1998)
19. d'Aquin, M., Sabou, M., Motta, E., Angeletou, S., Gridinoc, L., Lopez, V., Zablith, F.: What can be done with the semantic web? An overview of Watson-based applications. In: Proceedings of the Fifth Workshop on Semantic Web Applications and Perspectives (SWAP 2008), Rome (2008)
20. d'Aquin, M., Motta, E., Sabou, M., Angeletou, S., Gridinoc, L., Lopez, V., Guidi, D.: Toward a new generation of semantic web applications. *IEEE Intell. Syst.* **23**(3), 20–28 (2008)
21. Angeletou, S., Sabou, M., Specia, L., Motta, E.: Bridging the gap between folksonomies and the semantic web: an experience report. In: Proceedings of the International Workshop on Bridging the Gap between Semantic Web and Web 2.0 at the Fourth European Semantic Web Conference (ESWC 2007), Innsbruck (2007)
22. Peroni, S., Motta, E., d'Aquin, M.: Identifying key concepts in an ontology through the integration of cognitive principles with statistical and topological measures. In: Proceedings of the Third Asian Semantic Web Conference (ASWC 2008), Bangkok. Lecture Notes in Computer Science, vol. 5367, pp. 242–256. Springer, Berlin (2009)
23. d'Aquin, M., Motta, E., Sabou, M., et al.: Towards a new generation of semantic web applications. *IEEE Intell. Syst.* **23**(3), 20–28 (2008)
24. Sabou, M., d'Aquin, M., Motta, E.: Exploring the semantic web as background knowledge for ontology matching. *J. Data Semant. XI. Lecture Notes in Computer Science*, vol. 5383, pp. 156–190, doi: 10.1007/978-3-540-92148-6_6 (2008)
25. Tummarello, G., Cyganiak, R., Catasta, M., Danielczyk, S., Delbru, R., Decker, S.: Sigma: live views on the Web of Data. In: Demonstration at the Proceedings of the 19th World Wide Web Conference (WWW 2010), Raleigh (2010)
26. Tummarello, G., Oren, E., Delbru, R.: Sindice.com: weaving the open linked data. In: Proceedings of the Sixth International Semantic Web Conference and Second Asian Semantic Web Conference (ISWC/ASWC 2007), Busan. Lecture Notes in Computer Science, vol. 4825, pp. 547–560. Springer, Berlin (2007)
27. Finin, T.W., Sachs, J., Parr, C.S.: Finding data, knowledge, and answers on the semantic web. In: Proceedings of the 20th International Florida Artificial Intelligence Research Society Conference (FLAIRS 2007), Key West, pp. 2–7. AAAI, Menlo Park (2007)
28. Haase, P., Stojanovic, L.: Consistent evolution of OWL ontologies. In: Proceedings of the Second European Semantic Web Conference (ESWC 2005), Heraklion. Lecture Notes in Computer Science, vol. 3532, pp. 182–197. Springer, Berlin (2005)
29. Zablith, F.: Dynamic ontology evolution. In: Proceedings of the Seventh International Semantic Web Conference (ISWC 2008), Doctoral Consortium, Karlsruhe (2008)

30. Zablith, F., Sabou, M., d'Aquin, M., Motta, E.: Using background knowledge for ontology evolution. In: Proceedings of the Second International Workshop on Ontology Dynamics (IWOD 2008) Co-located with Seventh International Semantic Web Conference (ISWC 2008), Karlsruhe (2008)
31. Fernández, S., Berrueta, D., Shi, L., Labra, J.E., Ordóñez de Pablos, P.: Mailing lists and social semantic web. In: Patricia Ordóñez de Pablos Miltiadis D. Lytras (ed.) *Social Web Evolution: Integrating Semantic Applications and Web 2.0 Technologies*. Social Computing: Concepts, Methodologies, Tools, and Applications Editor (s): Subhasish Dasgupta (George Washington University, USA), pp. 335–349 (2010)
32. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: a framework and graphical development environment for robust NLP tools and applications. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002), Philadelphia (2002)
33. Lopez, V., Sabou, M., Motta, E.: PowerMap: mapping the real semantic web on the fly. In: Proceedings of the International Semantic Web Conference (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 414–427. Springer, Berlin (2006)
34. Angeletou, S., Sabou, M., Motta, E.: Semantically enriching folksonomies with FLOR. In: Proceedings of the First International Workshop on Collective Semantics: Collective Intelligence and the Semantic Web (CISWeb 2008), Tenerife (2008)
35. Ding, L., Finin, T.: Characterizing the semantic web on the web. In: Proceedings of the Fifth International Semantic Web Conference (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 242–257. Springer, Berlin (2006)
36. Lawrence, S., Lee Giles, C.: Accessibility of information on the web. *Nature* **400**, 107–109 (1999)
37. Bharat, K., Broder, A.: A technique for measuring the relative size and overlap of public web search engines. In: Proceedings of the Seventh International Conference on World Wide Web (WWW 1998), Brisbane, pp. 379–388 (1998)
38. d'Aquin, M., Baldassarre, C., Gridinoc, L., Angeletou, S., Sabou, M., Motta, E.: Characterizing knowledge on the semantic web with Watson. In: Workshop on Evaluation of Ontologies and Ontology-Based Tools (EON 2007), Madrid (2007)
39. Wang, T.D., Parsia, B., Hendler, J.: A survey of the web ontology landscape. In: Proceedings of the Fifth International Semantic Web Conference, (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 682–694. Springer, Berlin (2006)
40. Allocca, C., d'Aquin, M., Motta, E.: Detecting different versions of ontologies in large ontology repositories. In: Proceedings of the Third International Workshop on Ontology Dynamics (IWOD 2009), Washington, DC. CEUR-WS Online Proceedings, vol. 519 (2009)
41. d'Aquin, M., Allocca, C., Motta, E.: A platform for semantic web studies. In: Web Science Conference (WebSci 2010), Poster Session, Raleigh (2010)
42. d'Aquin, M.: Formally measuring agreement and disagreement in ontologies. In: Proceedings of the Fifth International Conference on Knowledge Capture (K-CAP 2009), Los Angeles (2009)
43. Cheng, G., Ge, W., Qu, Y.: Falcons: searching and browsing entities on the semantic web. In: Proceedings of the 17th International Conference on World Wide Web (WWW 2008), Beijing, pp. 1101–1102. ACM, New York (2008)
44. Harth, A., Hogan, A., Delbru, R., Umbrich, J., O'Riain, S., Decker, S.: SWSE: Answers before links! In: Proceedings of the Semantic Web Challenge, CEUR Workshop Proceedings, vol. 295 (2007)
45. Buitelaar, P., Eigner, T., Declerck, T.: Ontoselect: a dynamic ontology library with support for ontology selection. In: Proceedings of the Demo Session at the Third International Semantic Web Conference (ISWC 2004), Hiroshima (2004)
46. Thomas, E., Pan, J.Z., Sleeman, D.H.: Ontosearch2: searching ontologies semantically. In: Proceedings of the Third International Workshop on OWL: Experiences and Directions (OWLED 2007), Innsbruck. CEUR Workshop Proceedings, vol. 258 (2007)
47. Ungrangsi, R., Anutariya, C., Wuwongse, V.: SQORE-based ontology retrieval system. In: Proceedings of the 18th International Conference on Database and Expert Systems Applications (DEXA 2007), Regensburg. Lecture Notes in Computer Science, vol. 4653, pp. 720–729. Springer, Berlin (2007)
48. Myers, P., Espinosa, R., Parr, C.S., Jones, T., Hammond, G.S., Dewey, T. A.: The Animal Diversity Web (online). <http://animaldiversity.org> (2006). Accessed 5 Aug 2009

17 eScience

Jun Zhao¹ · Oscar Corcho² · Paolo Missier³ · Khalid Belhajjame³ ·
David Newmann⁴ · David de Roure⁵ · Carole A. Goble³

¹Oxford University, Oxford, UK

²Universidad Politécnica de Madrid, Boadilla del Monte, Madrid,
Spain

³University of Manchester, Manchester, UK

⁴University of Southampton, Southampton, UK

⁵University of Oxford, Oxford, UK

17.1	<i>Introduction</i>	703
17.2	<i>Emerging New Science</i>	704
17.2.1	An Example of a Scientific Research Scenario	704
17.2.2	Features and Challenges of This New Approach to Science	705
17.2.3	eScience for Scientists	706
17.3	<i>eScience Needs and the Role of Semantics in eScience</i>	707
17.4	<i>Example Applications</i>	710
17.4.1	Management of Research Data: The Virtual Solar-Terrestrial Observatory and the Semantically Enabled QUARC System	711
17.4.2	Support for Multidisciplinary Collaborative Research: The myExperiment Social Virtual Research Environment	714
17.4.3	Reuse, Reinterpretation, and Quality of Experiments: The Qurator Workbench	716
17.4.4	Data and Tool Discovery: The BioCatalogue Web Service Registry and the QuASAR Web Service Semantic Annotation	718
17.4.5	Data Cross-Linking of Data Sources: Scientific Linked Open Data and SensorGrid4Env	721
17.4.6	Stopping the Division Between Data, Experiments, and Scientific Publishing: Prospect and Semantic Publications	725
17.5	<i>Related Resources</i>	727
17.5.1	Management of Large Volumes of Research Data	728
17.5.2	Support for Multidisciplinary Research	728
17.5.3	Knowledge Reuse and Reinterpretation	729
17.5.4	Search and Discover the Right Data and Tools	729

17.5.5	Cross-Linking of Data	730
17.5.6	Stopping the Division Between Data, Experiments, and Scientific Publishing	731
17.6	<i>Conclusions and Future Issues</i>	731
17.7	<i>Cross-References</i>	733

Abstract: This chapter looks into how the use of semantic technologies can provide support to common needs in eScience projects, including data-intensive science, facilitating experiment knowledge reuse and recycle among scientists, lowering the barriers of knowledge exchange for interdisciplinary research, and bridging the gap between data from different sources and the gap between data sharing and digital scholarly publication. To illustrate this, we describe a set of pioneering semantic eScience projects that cover a diversity of application domains including bioinformatics, biology, chemistry, physics, environmental science, and astronomy, and we summarize some of the open issues and future lines of research and development in this area.

17.1 Introduction

The fast advances of technologies transform the way scientific research is performed. Data analysis and storage has moved from a paper-based, manual affair to an activity in which computers are vital. As a result, a vast amount of scientific data is being daily collected or produced by computational equipment. For example, to date, there are over 1,000 bioinformatics databases, reported in the annual databases issue of *Nucleic Acids Research* [1]. No single research organization has enough resources to collect everything; hence, the data-gathering and archiving processes are distributed and scattered at different places. Neither does any single research group have the computational power to process all these data; hence, scientists have to divide their analysis tasks into smaller jobs and distribute them to other available computers on their organization clusters or on the Internet [2]. Besides, collaboration among scientists from different institutions or disciplines is necessary in many occasions to apply a spectrum of methods and models to analyze and process this deluge of information, and the ability to access and reuse datasets, methods, models, and results of existing scholarly publications generally ensures more effectiveness and better quality in the research that can be carried out.

The development of eScience is a response to these emerging trends in scientific research. eScience was originally conceived as the application of computing to traditional science (mostly empirical, although in some cases theoretical as well) in order to empower scientists with their research in traditional activities such as modeling, simulation, and prediction, among others. However, now eScience can be considered to have gone further than that, and is even being considered as a third leg of the scientific method, together with the theoretical and empirical ones, by introducing a new environment in scientific research that has also led to new research methods that may potentially lead to better science. One of these changes in the scientific research conditions is due to the large volume of research data generated by distributed eScience research centers across multiple disciplines, which means that now no individual scientist possesses knowledge of sufficient depth to understand every aspect of the data being generated and archived. For example, in life sciences, the advent of mass sequencing and high-throughput experimentation such as transcriptomics, proteomics, and the other “omics” has drastically changed the landscape of biomedical research data; not only are there so many facts for any one scientist to know in his/her domain in

a comprehensive manner, but the availability of data from other domains that are pertinent to the scientist's domain also requires scientists to be knowledgeable in multiple domain languages. These urgent needs for interpretation and communication are not unique to life sciences; scientists from other disciplines face similar issues when trying to interpret numeric values and theoretical formulas shared with them by their remote collaborators.

Giving support to some of these new requirements arising from this new approach to science requires in some cases the explicit definition of the meaning of data about these different domains. This is the role that explicit semantics and their associated technologies, models, and methods can play, in the context of what is known as *semantic eScience*. That is, while traditionally eScience has mainly addressed issues of data and computation distribution, interoperation, and high performance in traditional and nontraditional scientific research tasks, the main focus of semantic eScience is on the application of explicit semantics over the eScience infrastructure to drive more accurate information interpretation, more efficient scientific analyses, and better collaboration among scientists, among others. This will be the main focus of what we present in this chapter.

The rest of this chapter is organized as follows. We first analyze and present the needs and opportunities that arise from the combination of semantics and eScience to empower the next generation of data-intensive cross-interdisciplinary science. Then we show various example projects from within semantic eScience that apply semantics in different ways with the objective to support different scientific disciplines, including bioinformatics, biology, chemistry, environmental science, and physics. The following section lists a set of reference resources that are related somehow to the development of semantic eScience, sharing and contributing to this vision, but that have not been covered in detail in this chapter. Finally, we summarize by reflecting on the commonalities and differences among the example projects that have been presented, on how they do and how they do not address the existing challenges that are identified at the beginning of this chapter, and on why some of these semantic eScience projects are not yet in a production stage.

17.2 Emerging New Science

This section starts with the description of a modern scientific research scenario that aims at illustrating the emerging trends in scientific research and their requirements to computing technologies, and vice versa (the advances in computing technologies that translate into changes and into scientific research). It then presents where eScience can help with some of the requirements and where it cannot or where it falls short to provide a good solution to those requirements, and what supports the need to combine semantics and eScience.

17.2.1 An Example of a Scientific Research Scenario

Paul, a postdoctoral researcher in a biology laboratory in the UK, is one of the team members of an ad hoc, worldwide consortium who together are investigating the

resistance of breeds of African cattle to a dangerous blood parasite. Experimentalists in Africa have produced gene expression data from resistant and nonresistant breeds. These genotype data are statistically analyzed by a team in the USA and combined with phenotypic trait data, metabolic pathway data, and sequence data available in in-house collections and public datasets of several model organisms, provided by major and minor organizations, which act as data providers.

As part of his normal research activity, Paul searches the published literature for common candidate genes that appear in the phenotype literature, and obtains several promising leads from this search. However, Paul not only works proactively in the identification of these leads: a news syndication feed to which Paul is subscribed also prompts Paul with newly published data by a reputable lab on a related parasite in mice. With these results, and especially with the availability of these published datasets, Paul can start different activities or pipelines of data comparison and analysis.

Therefore, for the time being, Paul has been able to obtain a set of facts and to propose some conjectures. Besides, Paul has identified good reusable datasets and has pooled some literature that can be analyzed for finding patterns and relationships among the facts obtained. Then a set of hypotheses emerge (either from a direct analysis made by Paul or any of his colleagues on these sets of analyses, or because similar facts, patterns, and/or relationships are found in public or private libraries of already solved cases). For example, one of the possible hypotheses can be that a gene related to cholesterol regulation is also implicated in trauma survival in intensive care patients. Following a lab-developed standard operating procedure, the experimentalists re-sequence the gene in order to identify polymorphisms that might correlate with the phenotype.

The computational and data analysis procedures are fully documented in Paul's lab book, which is available as a combination of handwritten notes and well-documented electronic information. As a result of this research, Paul and his colleagues produce a set of peer-reviewed articles, published together with the original and resulting datasets and their annotations, which are then linked to the publications and to the protocol followed.

17.2.2 Features and Challenges of This New Approach to Science

The previous example demonstrates a mainstream collaborative scientific research scenario that raises new challenges to computing and collaboration technologies.

First, Paul does not work on his own or on the restricted scope of his research group. He collaborates with multiple institutions worldwide and performs interdisciplinary research. As a result of the extended scope and collaboration context, he needs support on *interpreting the data* shared by his collaborators, and on *searching and analyzing the distributed datasets* as if they were available in a single centralized store.

Paul must be able to search for useful information quickly from the growing number of genomic data resources, using a variety of bioinformatics tools. At the same time, he must be able to manage the data generated in his local lab and in his colleagues'

lab and *cross-link them with the public knowledge bases* in order to find patterns and draw conclusions. New research hypotheses might be produced in this process, which will lead to more experiments in the lab and new research datasets. He must also be able to *record the experimental process* in order to repeat the experiments and to provide evidence for his results.

More research data are being shared in this collaborative science environment, either within a small group of collaborators or through scientific publications. Paul may have different levels of *trust on the data* that are shared by research groups or people, depending on whether he knows them or has collaborated with them before, or on their overall reputation in the scientific community. Besides, he needs an effective means to *retrieve research data or re-execute experiments that are published in articles*, without the need to regenerate the data, the code, or the workflows used to derive the results presented in these publications.

17.2.3 eScience for Scientists

The previous example shows that with eScience moving toward interdisciplinary research, the need for accessing and integrating data resources collected by individual research groups becomes even more pressing [3]. These distributed resources may contain observational data collected using instruments with varying settings and containing measurements captured in a wide range of formats. A researcher can no longer possess knowledge of sufficient depth to understand the entire data collection [4]. These data sources need to be presented to the scientists as if they are from one store and as if they are available in a homogeneous format, so as to facilitate their manipulation. Besides, the meaning of the numeric values, theoretical models, and analysis programs must be well defined so that the scientists can interpret the information integrated from the distributed sources without being an expert in every subject.

Given the scale of data sources and the complexity of analytical processes used in this type of research, the automation of the experiment would provide clear benefits. This can be achieved, for instance, by means of scientific workflows executed in workflow workbenches [5], which combine and orchestrate data and services, representing a step toward the “industrial scale” science to cope with industrial-scale datasets. To allow this use of data sources, these need to become programmatically accessible through computing technologies (e.g., Web Services [6] or Representation State Transfer (REST) approaches).

Not only should the process of designing and running experiments be automated but also an accurate recording of the experiment execution (provenance [7]) should be maintained. Provenance information enables scientists to perform reproducible eScience experiments; the running of workflows can be aided by the results of earlier runs. Experiments can be repeatedly executed to identify and characterize any new features, and this information also increases credibility of research outcomes. In addition, analysis of provenance logs enables smart runs, resuming workflows from a given point during their past execution, and evaluation of the quality of services.

17.3 eScience Needs and the Role of Semantics in eScience

eScience empowers the execution of data-intensive experimentation, as described above. However, to fully interpret their research results and draw scientific conclusions from them, scientists need to seamlessly blend their own results with public databases and research data reported in journal publications or sent to them by their collaborators. *Local data results* must be managed and documented with sufficient structured metadata to facilitate efficient search and retrieval in a local context. *External public data* must be accessible in a format interoperable with their local data for efficient data integration and be documented using a controlled vocabulary that they all understand and that accurately reflects the meaning of the data.

The following paragraphs describe more specific needs that can be derived from eScience.

The need to manage the continuously growing amount of research data: Data-intensive science means that data have to be organized, indexed, and formatted. Often it is not known what types of data will be collected and what new datatypes are coming along. Therefore, there is a need to use open flexible data models to avoid fixed formats and premature structuring that would lead to data tombs.

The need to support the multidisciplinary collaborative nature of science: To support the collaborative team (experimentalists, informaticians, modelers, statisticians, and so on) working together requires data and methods to be explicitly described so that they can be sufficiently understood at touch-points by those outside the originating discipline, and this means exposing assumptions.

The need to reuse and reinterpret data, and to understand its quality: In Paul's team, data are reused and reinterpreted in ways unanticipated by their originators, often years after the data were produced. This point and the previous two mean that descriptive and accurate metadata is crucial, and that metadata must be controlled if they are to be accurately interpreted by colleagues, strangers, and future scientists. The provenance and quality of data is crucial for scientists to trace the originators of the data and establish trust of the data.

The need to discover data and tools best meeting scientists needs: Not only the number of data sources grows exponentially in the computer-aided sciences but also the number and diversity of the tools and applications. Apart from learning from their friends the useful tools and resources, scientists should also be provided with toolkits searching for resources best fitting their needs. This requires these resources to be described in a domain language that is familiar to the scientists, and simple and effective for discovery.

The need to cross-link, integrate, interoperate, aggregate, and compare new and legacy data: The need to cross-link, integrate, interoperate, aggregate, and compare new and legacy data from different disciplines, from different groups, in different formats, collected under different experimental conditions for different purposes that were not designed to be combined from the outset can also be derived. This means that data have to be systematically identified, provenance has to be systematically collected, controlled vocabularies have to be adopted to interlink metadata content and data results, and a common format has to be enforced for exchange.

The need to stop the division between data, experiments, and scientific publishing: Two new publications are submitted to PubMed every hour. The materials, methods, and results of science are seamlessly blended and interrelated. Common identifiers must be provided for people and data, and controlled vocabularies and metadata must be established for cross-linking and integration, in fact, not only text but also data and experiments that can be enacted.

Semantic technologies have the potential to support most of the aforementioned needs. On the one hand, semantic technologies and standards the achievement of an interoperable representation of data and the seamless integration of data from different sources. They also provide the languages for expressing the meaning of resources (data, information, documents, links, etc.) in a machine-processable way. Together, these two aspects facilitate the sharing of data and allow their accurate interpretation [8] when they are passed between different communities of different background or levels of expertise.

Let us see now in more detail how the previously identified needs can be supported by semantic technologies.

To organize and index the continuously growing amounts of research data: Accurate and structured metadata are needed to describe their content, meaning, and provenance. Resource Description Framework (RDF) [9] is to data what HyperText Markup Language (HTML) is to documents, permitting us to describe resources (such as data, researchers, and experiments) on the Web.

To support multidisciplinary collaborative research: Data must be made accessible and be accompanied with accurate descriptions about their meanings. Ontologies [10] provide controlled vocabularies for the scientists to explicitly express their interpretation about data and to exchange their interpretations with their colleagues in a common, well-defined language. They can be defined using RDF Schema (RDFS) [11], which is a schema language that can be used to define simple taxonomies supporting subsumption reasoning, or using the Web Ontology Language (OWL) [12], which is a family of expressive description logic-based ontology languages that can express and constrain knowledge models and support more complex description logic-based reasoning.

To support reuse and the reinterpretation of experiment resources, and the analysis of its quality: Not only explicit, accurate metadata are needed to describe the data resources and services that are used and about the execution of experiments (provenance) but also a framework to bring trust to scientists and facilitate their collaboration. Scientists often have their own set of measurements for evaluating the quality and trustworthiness of data. This knowledge can be captured using ontologies in order to replicate and automate the evaluation process using machines, as it happens with provenance models [7]. The relationship between people in the online social network can be described using vocabularies such as Friend of a Friend (FOAF) [13] and Semantically Interlinked Online Community (SIOC) [14], to name a few. All this information can be combined in order to achieve a better understanding of the experiments and the collaboration between scientists.

To search for right data resources and tools for scientists: Not only are RDF and ontologies needed to provide structured metadata about the data and tools but also

query languages and search algorithms to discover information from the large volume of RDF data. The SPARQL query language and protocol [15, 16] allows the execution of queries over RDF data. Data providers could make their datasets available through SPARQL endpoints, permitting a dataset to be programmatically accessed by other applications using SPARQL queries.

To cross-link data from different sources: To cross-link data from different sources including data from legacy databases, experiments, or digital publishing, new applications should use the Web as the platform to create a web-scale database. The Web has been proven to be a scalable platform for sharing documents; its decentralized infrastructure allows for large-scale information dissemination and linking [17], and has the potential for disseminating the fast-growing research data [18]. This is known as the Web of (linked) data [19], promoting the publication of data on the Web using new Web standards and evolving the Web as a global information space of both documents and data. Two key Web technologies lay the foundation for building the Web of Data: the HyperText Transfer Protocol (HTTP) [20], which permits the transportation of information resources on the Web, and the Uniform Resource Identifiers (URIs) [21], which provide a globally scoped system for identifying Web resources unambiguously. More on the Web of Data can be found in [Semantic Annotation and Retrieval: Web of Data](#).

To stop the division between data, experiments, and scientific publishing: This requires not only structured metadata about data, experiments, and papers but also the ability of providing links among them. Workflow workbenches are an example of those tools that can be used in scientific publications to embed and replicate experiments, and to use and generate data, and RDF data and ontologies can be used to provide common identifiers for people and data, controlled vocabularies, and metadata for cross-linking and integration. As a result, a publication may stop being a document-oriented artifact.

Although these standards and technologies lower the barriers for building applications supporting the new science, there are still many technical and social issues that must be addressed to bridge the gap between computing technologies and the real needs from scientists. Besides, semantic technologies should not be considered as the Holy Grail for eScience. Some of the current limitations that we can identify in the previous topics are the following:

Organization and indexing of growing amounts of research data: While RDF is a flexible model for the description of any type of resources, which is based on the use of triples as a basic mechanism for the representation of their properties, it is not the most adequate representation mechanism for some types of data that are common in eScience, such as numerical values, data streams, and spatiotemporal information, for which there are specialized representation mechanisms that have been used traditionally in scientific domains.

Support of multidisciplinary collaborative research: Although we have pointed out that ontologies are the artifacts that allow the formal explicit description and exchange of interpretations, and although there are different languages to express them (RDFS and OWL, and in the latter case different profiles), they may be again not suitable for expressing some of the constraints that we may need to use when dealing with scientific data. For instance, the representation of probabilistic or non-monotonic knowledge (e.g., this concept

can be defined in this manner in 98% of situations, but there are situations where this is not valid) is currently out of the scope of the expressivity features of these languages. This is also the case with some type of rule-based knowledge, trust management, etc.

Support for the reuse and reinterpretation of experiment resources, and the analysis of its quality: Provenance models, social network models, etc. allow characterizing the metadata that can be attached to specific data resources. However, there is a need to go deeper into models for the derivation of quality and trust metrics of data based on this type of information.

Search for data resources and tools: In this sense, although SPARQL is useful as a query language for ontology-based metadata, which is key in the eScience domain, there are additional search needs in this context that are not covered yet by the current specifications, such as the handling of data streams, aggregate functions, or statistical models, although there are works related to this area, proposing extensions to support these functions. Besides there is a need to improve the distribution of queries to distributed RDF datasets, and to improve the efficiency of the query-answering algorithms.

Cross-linking of data from different sources: The linked data approach for the publication and retrieval of metadata has proven useful in a number of domains, also outside the scientific context. Given this recent success, there is now a great possibility to cross-link and relate data from different sources, although there are still open issues in terms of discovering URIs that can provide metadata, exploiting queries over the Web of linked data, determining the trust and quality of this data, etc.

Relationship between data, experiments, and scientific publishing: There is still a lot of work to do, both technologically and socially, in order to strengthen the relationships between data and experiments and the way that they are published in the scientific context, going further than the current paper-based model where datasets and experiment workflows are only described but not provided together with the description of the results obtained with them, so as to make them reusable.

In summary, the aim of this section has been to show that there are a number of challenges and opportunities associated with the use of semantics in eScience, and that several of them have to be overcome in order to provide full support to the eScience life cycle (including the life cycle of experiments and of data resources). In the following sections, we will describe how some of these challenges have been overcome in the context of specific applications that can be considered as exemplar eScience applications.

17.4 Example Applications

This section presents a selected collection of state-of-the-art semantic eScience projects, demonstrating how they apply semantics successfully for the construction of applications that best meet the scientists' needs and how they address some of the key issues identified in [▶ Sect. 17.3](#). This list of applications does not aim at being exhaustive but only representative. Besides, although these projects and applications are organized according to the needs that we identified in [▶ Sect. 17.3](#), this does not mean that they do not also

cover some of the other identified needs. However, they have been selected according to their core features and are described mainly according to the selected topic.

17.4.1 Management of Research Data: The Virtual Solar-Terrestrial Observatory and the Semantically Enabled QUARC System

The Virtual Solar-Terrestrial Observatory (VSTO) demonstrates how semantic technologies can be applied to provide scientists access to observational datasets from a spectrum of scientific disciplines, spanning from atmospheric terrestrial physics to solar physics [4].

The VSTO data portal is one of the main entry points to this effort, and is underpinned by a background network of ontologies that captures key knowledge across several scientific disciplines, and which is organized according to six main areas: instruments, observatories, operating modes, parameters, coordinates, and data archives [22]. The aim of using this ontology network is mainly because it allows the reduction of the ambiguity of similar terms used for different meanings in different disciplines, and because it allows the unification of multiple terms for the same phenomenon or process, again across disciplines.

The design of the ontology network is driven by use cases, which start by identifying the breadth and depth of the science terms that are required to express the use cases. The development of these ontologies followed a traditional ontology development methodological approach, where existing ontologies in these domains were surveyed, reused, and extended to cover in-depth the needed knowledge. The development process started by focusing on those areas that had potentially higher impact, such as the instrument ontology, since it would be more reusable across domains. This ontology contains terms and definitions such as “a *spectrophotometer* is a *photometer* which is an *optical instrument*.” For example, using the knowledge encoded in the previous definition, the VSTO data portal can return data from the *Davis Antarctica Spectrometer*, which is classified as a *spectrophotometer*, to scientists looking for *photometric* data; even though these scientists have no knowledge about the kind of data the *Davis Antarctica Spectrometer* can produce [4].

To provide reliable scalability, VSTO uses semantically enabled Web Services to find, retrieve, and manipulate data from each source repository [23]. Considering the overwhelming quantity of data that VSTO manages, using Web Services, to retrieve information directly from the source data repository performs far better than creating a central knowledge repository containing all the information, and at the same time provides better means to manage the distribution of information.

Another important functionality of the data portal is that it allows scientists to form syntactic and semantic queries to distributed data sources as if they are organized, stored, retrieved, and queried in a common schema and format. This means that users need no longer to learn the schema and query language of each source repository, since all the information available in the system is now presented to these scientists as being described in one controlled vocabulary. As a result, it becomes easier for these scientists to form

meaningful and correct queries without having to have very deep knowledge across different subjects, which allows a better exploratory analysis of the data available in the system. For example, scientists can ask for photometers even if some of the data are not obviously available as photometers by their original names in their original sources.

Finally, as an interesting aspect that is worth commenting about the VSTO initiative, user needs are constantly changing, which requires a continuous maintenance of the background ontology networks, so as to retain the functionality of the data portal according to these changing needs. The ontology network is maintained through an iterative, incremental approach: when new use cases arise, the minimum number of classes and properties necessary to address those new use cases is created and the minimum number of associations and class value restrictions is introduced to the existing ontology in order to reduce unnecessary complexity of reasoning required for the use cases. Besides, keeping the ontology network as simple as possible also increases its opportunity of being reused in other contexts or applications [22].

Another good example of how semantic technologies, in general, and ontologies in particular are useful in the management of research data is the semantically enabled extension of the QUARC system [24] for the Envisat satellite [25], which is used in the context of Earth observation to monitor the evolution of environmental and climatic changes, and whose data facilitate the development of operational and commercial applications.

Earth observation can be defined as the science of getting data about our planet by placing in orbit a hardware/software element with several observation instruments, whose main goal is to obtain measurements from the Earth surface or the atmosphere. The instruments onboard the satellite act like cameras that can be programmed to take images of specific parts of the Earth at predefined times. These data are sent to ground stations and then processed in order to get meaningful scientific information.

Parameters for instrument operations and for the satellite configuration constitute the mission plans issued by the mission planning system. These plans are issued regularly (e.g., on a weekly basis), and can be modified until they are sent to the satellite. Catastrophic events such as earthquakes, volcanic eruptions, and hurricanes are examples of events that can cause last-minute replanning. These plans and their modifications are sent to the flight operation segment (FOS), which in turn resends that information to a ground station and from there to the satellite antenna of the spacecraft. A computer onboard the satellite stores the list of macrocommands (MCMD) that request an instrument or any other part of the satellite to perform an action. These macrocommands include loading a table, triggering an operation, and getting internal status information. Images from each of the instruments are stored onboard (in the satellite computer memory) as raw data and when the satellite overflies the ground station those data are sent to the ground station antenna (data downlink). Conversion from the raw data to higher level “products” (adding identification labels, geo-location data, etc.) is performed sequentially at the ground station and various payload data segment facilities.


The Envisat satellite carries ten different instruments. Data circulates within the system as various plan, macrocommand, and product files, with well-defined structures. There can be a variety of hardware or software problems that can occur within the process; hence, there

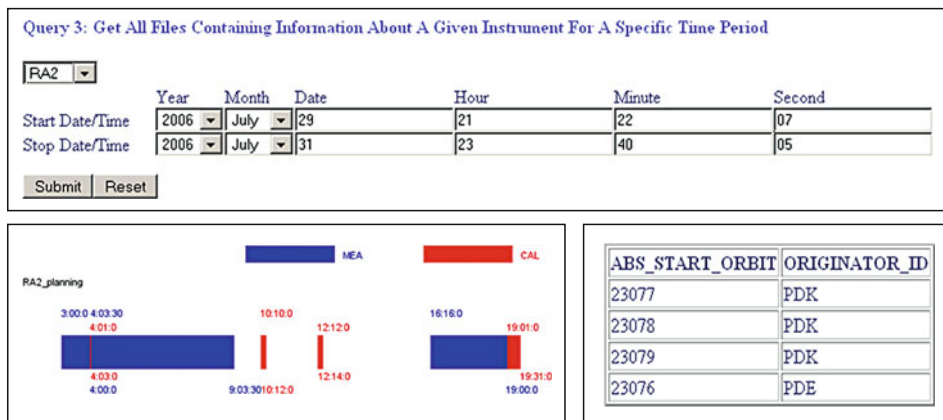
is a need for the system to be monitored. QUARC is a system that checks offline the overall data circulation process and in particular the quality of the instrument product files. It checks that the satellite and the instrument have performed the measurements successfully (taking images of the Earth), that these images have been stored onboard and transmitted as raw data to the ground station, and then that they are processed correctly. QUARC returns reports and plots, which help in the production of new plans. Additionally, the QUARC system is designed to assist in decision-making when an instrument or the whole system malfunctions and to detect, in a semi-automated fashion, that something incorrect has occurred in one part of the product generation or data circulation.

The operational QUARC system is located in a single location (ESA-ESRIN, in Italy), which communicates with the archive containing all the products generated from the beginning of the mission and with all the other facilities. The data ingestion modules, one per filetype, read the files and convert their contents into parameters that are meaningful to the QUARC data model. The system has been specifically built for this purpose and has bespoke user interfaces (for instance, it stores implicit metadata about the files that it manages hidden in long file names). It took several years to build it and there are significant maintenance and development costs as new reports are required and new missions are launched.

The objective of the semantically enabled extension of the QUARC system is to better enable the comparison between the planned activity and the production of data by the satellite and the further processing of those data in ground systems, demonstrating that greater degrees of flexibility, scalability, interoperability, and extensibility than the existing system can be achieved by the use of semantic technologies, together with a more efficient development of new functionalities.

Some of the main challenges of this system are in the extraction of implicit metadata used across the system. The existing QUARC system exposes these metadata through bespoke interfaces. For example, much of the metadata are encoded in specific, amalgamated identifiers, with “implicit semantics.” For example, rules had to be created for product filenames like “RA2_MW__1PNPDK20060201_120535_000000062044_00424_20518_0349.N1.” This is decomposed into an event type (RA2_MW), processing level (1P) and center (PDK), a sensing start time (2006-02-01:12.05.33), and so on. Generic metadata (applied across all captured metadata) and the ontology further add, for example, that the event type (RA2_MW) is executed by a particular instrument, the radar altimeter. Not only are these metadata extracted, but also some information about the provenance of this information, the life cycle of the extracted metadata, etc.

Once that all the implicit semantics has been extracted, made explicitly available in RDF format according to a set of ontologies about satellites, instruments, events, etc., and recorded in distributed RDF repositories, it can be exploited by QUARC users (who are in general Earth observation scientists), by means of flexible query interfaces that RDF provides.  [Figure 17.1](#) shows a sample interface that allows for the posing of queries about the information of a given instrument for a specific time period. This form-based query is actually transformed into SPARQL and executed over the distributed RDF repository that holds all the extracted metadata.



■ Fig. 17.1

A sample RDF query interface for QUARC-related data

17.4.2 Support for Multidisciplinary Collaborative Research: The myExperiment Social Virtual Research Environment

myExperiment was first envisioned as a social networking site for scientists, where they could make friends and join groups much like it is possible to do on general-purpose social networking platforms like Facebook or MySpace [26]. The main difference of myExperiment with respect to these general-purpose platforms was to tailor the Website for use by scientists, allowing them to share their research (including topics, results, and data sources) with colleagues and distance collaborators. In the first instance, this was to support the domain of bioinformaticians, so that they could share their workflows allowing others to reuse or adapt them for their own purposes and hence saving significant development time [27].

Over time, myExperiment has come to support other user groups that are not necessarily related to bioinformatics or only use workflows as one of their core Research Objects (ROs), and has realized that what is shared is not necessarily just a single file but multiple interlinked files, for example, a workflow with various graphical representations, inputs and outputs, and documents explaining usage. It was also observed that the metadata for a file and annotations (e.g., tags, comments, and ratings) added by users were also important aspects of what was being shared (encapsulated in what is called an RO). As a result of these observations and of the subsequent development activities around the platform, myExperiment has evolved into what could be defined as a Web-based social virtual research environment (Social VRE) [28], with three main features: content management, social networking, and object annotation.

Users submit contributions (from file uploads to more abstract concepts) along with supporting metadata, in what is called an RO. As part of the process of submitting

a contribution, a user defines a customizable usage policy (a license schema) so that this contribution can be shared with other users. As part of the contribution submission, users from inside or outside myExperiment can be credited for their contributions, and contributions can also aggregate interrelated resources, in what is known as a pack contribution. A pack allows both items local to myExperiment as well as external resources to be grouped together, and can be constructed by a single user or collaboratively by multiple users.

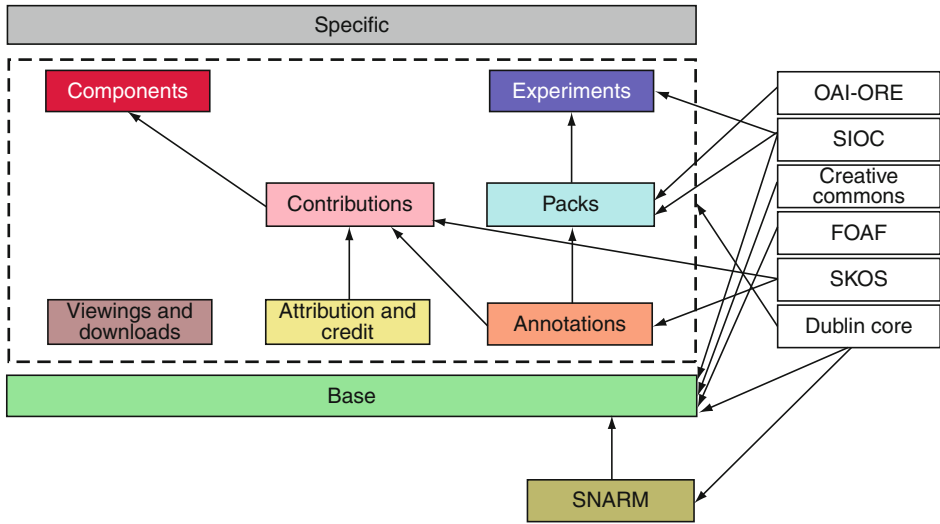
Users can also build up their social network by becoming friends with other users and by joining relevant groups (e.g., for the project they are working on, the physical research group they belong to, or just with users who share similar interests). These friendships and memberships are then provided as options to the user when they want to share their contributions. Users can choose to keep a contribution private, share it with friends or specific groups, or make it completely public.

Once a contribution is available on myExperiment, permitted users can add annotations to it, which form part of their associated metadata. These annotations include tags, ratings, comments, reviews, citations, and favorites. Annotating objects enhances searching and supports community curation, making it easier for all users to find the contributions they are interested in and then determine whether they are useful.

The myExperiment data model is expressed using RDF, according to the OWL myExperiment ontology (<http://rdf.myexperiment.org/ontologies/>). This ontology was designed with the intention of making it as reusable as possible by similar projects:

- First, the ontology reuses terms from well-known core schemas/ontologies such as Dublin Core, FOAF, SIOC, and Open Archives Initiative's Object Reuse and Exchange (OAI-ORE). Through this, the ontology and the RDF it specifies are given a graceful degradation of understanding [29].
- Second, the ontology is broken down into a set of modules (or a set of ontologies if we consider the myExperiment ontology as an ontology network, see ► [Fig. 17.2](#)). These modules consist of a “base” module that defines the core aspects of myExperiment, namely, *content management*, *social networking*, and *object annotation*. Additional modules define types of contributions, types of annotations, credit, packs, experiments, usage statistics, and workflow components. Each module reuses terms from core schemas/ontologies where appropriate, including the ones identified above.

Finally, the goal of ROs is to provide a representation for “reproducible research,” which is a key part of scientific research. An RO should capture sufficient information to allow a research process to be repeated either by the scientist who originally created it or by anyone who has access to the RO, granted by the scientist who uploaded it. By capturing these detailed ROs it should also be possible to facilitate adaptation and re-purposing so they can be applied to different scenarios or to provide plans for similar experiments/processes.



■ Fig. 17.2

myExperiment ontology modules architecture

17.4.3 Reuse, Reinterpretation, and Quality of Experiments: The Qurator Workbench

The motivation of the Qurator project comes from the observation that, in several practical application domains and in experimental eScience in particular, information consumers need to make informed decisions regarding the acceptability or “fitness for purpose” of information that is required to carry out their experiments, or of the experiment’s outcomes. Not only explicit objective community-shared knowledge is used to reach such decisions, but often latent, implicit personal knowledge (also known as *quality knowledge*) is used during the decision process. The informal nature of quality knowledge, however, has two main drawbacks. First, it makes it difficult to document the decision process to the extent needed to incorporate it into an otherwise disciplined experimental process, and thus to make it *repeatable*, an essential requirement in experimental science. And second, the quality criteria used by a scientist within a community and in a given experimental context are often potentially reusable by others in the same field of science. However, implicit quality knowledge makes reusability hard to achieve. These considerations made the definition of a semantic model for the explicit representation of quality knowledge a priority for the project. In turn, such a semantic model made it possible to explore ways to make experimental eScience *quality-aware* with relatively low human effort, and in a way that scales up to large areas of science, through reuse.

In summary, in the Qurator project semantic modeling is used to formalize notions of quality of information in the context of eScience, and to drive the specification of formal processes, namely, scientific workflows, which encode the computation of user-defined quality metrics.

More specifically, the Qurator project worked on the hypothesis that all quality knowledge can be conveniently specified, in general, in terms of processes that classify (e.g., accept or reject) elements of an input dataset, based on user-defined quality criteria. Furthermore, each of these processes shares a common model, called a *Quality View* (QV), and its specific components are semantically annotated using an original information quality OWL ontology. The ontology consists of an upper model, immutable and domain-independent, and an extensible, domain-specific lower model. A simplified version is sketched in [Fig. 17.3](#), where the OWL axioms as well as a number of ancillary classes are omitted for simplicity. Here only a brief overview of the model is given, to show how OWL axioms are used to define constraints on QV processes. More details can be found in [30, 31].

The upper model defines a hierarchy of abstract concepts: A *data entity*, representing the top class for any concrete datatype, is the input to the QV process. A quality judgment on a data entity, denoted by class *Quality Assertion*, is a function that uses underlying objective indicators, represented by the class *Quality Evidence*, to classify the entity according to a predefined class structure, for example, “accept” and “reject.” In turn, constraints are imposed to enforce that there must exist some annotation function that computes the indicators. The classes just mentioned can be extended to capture domain-specific notions of data entities, indicators, annotation functions, and assertions, as exemplified in the middle part of [Fig. 17.3](#) for the case of a quality process for proteomics data (these are data produced by a protein identification algorithm, called *Imprint*). Ultimately, these class extensions along with the upper ontology provide a logical framework for describing a quality-based filter, which rejects protein identifications based on a rule that predicates on a number of specific indicators (such as “Hit Ratio” and “Mass Coverage”).

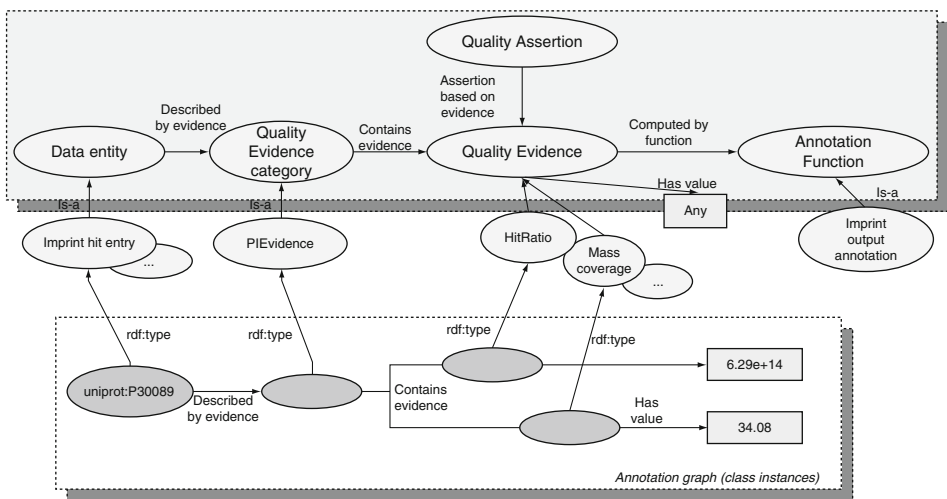



Fig. 17.3

Fragment of IQ ontology in Qurator

Finally, the bottom part of  Fig. 17.3 shows actual data and indicator values, modeled as individuals of their corresponding classes.

The ontology is used for two main purposes. First, since each component in a QV has an associated semantic type, which is a class in the IQ ontology, the class axioms are used to formalize the notion of a *consistent* QV. For example, QV is inconsistent if it includes at least one type of quality evidence, which, according to the ontology, is not computed by any of the annotation functions that appear in QV. The work in [31] shows how standard OWL reasoning can be used effectively to validate the consistency of a QV, when its elements are annotated with ontological concepts.

The consistency enforcement idea is exploited in the *semantics-aware* Qurator visual interface, which is used by scientists to interactively design their own new, personalized QVs. In practice, ontology reasoning is used to assist users in their design, by incrementally suggesting new components that are consistent with those that are already part of the process, eventually providing the guarantee that the entire QV is indeed consistent with respect to the ontology.

The IQ ontology and its associated consistency constraints are at the core of the Qurator workbench, a suite of user-oriented tools for the specification of QVs and their automated translation into executable Taverna workflows [32].

In summary, it is clear that this project shares some of the motivations listed in the introductory section, namely, in its attempt to provide a shareable, formal, extensible, and practically useful model for a knowledge domain that cuts across most areas of eScience, namely, that of information quality assessment.

17.4.4 Data and Tool Discovery: The BioCatalogue Web Service Registry and the QuASAR Web Service Semantic Annotation

Web Services are a means for packaging existing data and computational resources in a form that is amenable for use and composition by third-party applications. They provide a well-defined programming interface to integrate components or tools into other components, tools, or applications. Software applications written in various programming languages and running on various platforms can use Web Services to exchange data over the Internet, overcoming their heterogeneity.

However, one of the main issues that hinders the wide adoption and use of Web Services is the difficulty of discovering the *appropriate* Web Service for a specific task, that is, the Web service that performs the analysis that the scientist is looking for. For example, the workflow workbench Taverna (<http://taverna.sourceforge.net/>) provides access to over 3,500 thousand Web Services that can be composed by scientists for constructing and enacting their *in silico* experiments. Manually browsing this list of Web Services can be time consuming, tedious, and in many cases frustrating for scientists who are trying to build their experiment workflows. This is a major obstacle to the users of these services

(biologists, bioinformaticians, and tool providers) and a frustration to service providers, whose services are unknown, unused, poorly used, or misused as a result.

Keyword search capabilities such as those provided by Universal Description Discovery and Integration (UDDI) are normally not sufficient when it comes to service discovery activities. Hence semantic annotations of Web Services have been proposed to assist in this service discovery task, and can be used to support users in composing workflows, both by suggesting operations that can meaningfully extend an incomplete workflow and by highlighting inappropriate operation selections [33].

With the above issues in mind, the BioCatalogue project [34] aims to build a Web service registry dedicated to the life science community. At the time of writing, BioCatalogue contains semantic descriptions for 700 Web Services. As well as the syntactic description of Web Services, BioCatalogue caters for their semantic description. Specifically, the functionality of a Web service and the semantic domain of its input and output parameters are described using ontological terms, which use terms coming from the ^{my}Grid ontology [35].

Other salient features of BioCatalogue include the following:

1. *Curation of service descriptions*: The bioinformatics domain is familiar with the idea of managing curated data as a prerequisite for data use and integration. BioCatalogue gives support to the service description curation process and provides enough information to scientists in order to determine the degree of quality of the service descriptions and the process that they have followed in order to be advertised in the catalog.
2. *Web service discovery*: BioCatalogue allows users to discover Web Services by means of a combination of the following mechanisms:
 - (a) *Keyword-based service retrieval*: This allows users to locate Web Services of interest by providing as input one or multiple keywords that provide some information about the service. This includes things such as the name of the service and its associated tags.
 - (b) *Advanced search capabilities*: Searches are managed according to additional information about the functionality of the target Web service, the kinds of data it takes as input and/or the format of the results it delivers, its reputation, reliability, and so forth.
 - (c) *Traditional “form-filling” search mechanisms*: These are coupled with a modern facet-based browsing to provide a “shopping” style Web interface *à la* Amazon.

Although the distribution of the effort of manual annotation of Web Services in services such as BioCatalogue aims at distributing the effort needed to perform the time-consuming process of Web service annotation, this task still demands deep domain knowledge from individual annotators, which are both scarce and expensive, as well as consistency of interpretation within annotation teams. Because of this, the rate at which existing services are annotated lags well behind the rate of development of new services, which leaves a good number of Web Services without any type of annotation or with very basic annotations, which are not enough for obtaining good results during the Web service discovery activities.

In order to improve this situation, the QuASAR project (<http://img.cs.manchester.ac.uk/quasar>) developed a method for learning semantic annotations of Web Services, and focused on those that make use of sources of information such as repositories of trusted data-driven workflows (e.g., those coming from the myExperiment platform). If a workflow is known to generate sensible results, then it must be the case that the operation parameters connected by the workflow are compatible with one another (at least to some degree). In this case, if one side of a data link is annotated, that information can be used to derive annotation information for the parameter on the other side of the link. A simple example is used to illustrate this idea. Consider the pair of workflows shown in Fig. 17.4. Both these workflows are intended to perform simple similarity searches over biological sequences. The first finds the most similar protein to the one specified in the input parameter. To do this, it retrieves the specified protein entry from the Uniprot database, runs the Blast algorithm to find similar proteins, and then extracts the protein with the highest similarity score from the resulting Blast report. The second workflow finds similar sequences to a given DNA sequence. It retrieves the DNA sequence from the DDBJ database, searches for similar sequences using Blast, and finally extracts the sequences of all matches from the Blast report.

The parameters of the Blast operation have not been annotated in any of them, while the parameters of the other operations have been already annotated. Since these are thoroughly tested workflows, their data links should all be compatible, and therefore the existing annotations can be used to infer some information about the annotations that the Blast operation ought to have. For example, the domain annotations of the workflow convey the information that the input of the Blast operation must be compatible with both ProteinSequence and DNASequences, and its output must be compatible with both ProteinSequenceAlignmentReport and SequenceAlignmentReport. Based on this information, the inference can be made that the semantic domain of the input of the Blast operation should be a superconcept of the concept obtained by the union of ProteinSequence and DNASequences; similarly, the semantic domain of the output of

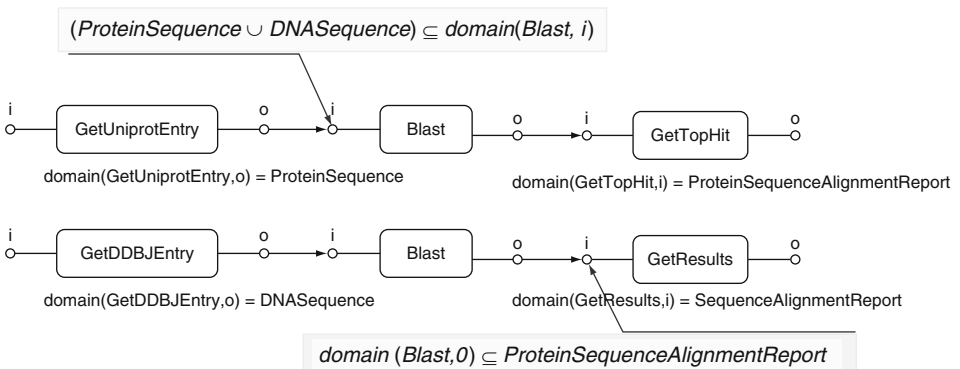


Fig. 17.4

Example workflows for simple similarity searches over biological sequences

Blast should be a subconcept of `ProteinSequenceAlignmentReport`, because `ProteinSequenceAlignmentReport` is a subconcept of `SequenceAlignmentReport`. The derivation mechanism has been implemented, and its practical applicability for inferring new annotations has been established through an experimental evaluation using real-world bioinformatics Web Services, as described in [36].

17.4.5 Data Cross-Linking of Data Sources: Scientific Linked Open Data and SemsorGrid4Env

Scientific data integration is a well-known hard problem [37, 38], independent of the scientific domain where it is applied. The main reason for this is that information is scattered in different databases, exposed through diverse user interfaces and access methods, and highly heterogeneous content-wise. Data content from different databases often overlaps, but information is represented in a diversity of formats and data are identified by heterogeneous identifiers. As a result, the attention of the scientists is often diverted from scientific innovation to the management of this multifold complexity. The interoperability and cross-links between databases must be addressed to provide the scientists better access to the distributed knowledge.

Traditionally, scientists rely on their notebooks to gather knowledge from different databases or copy-and-paste data analysis results from one Web page to another in order to follow a workflow of data management. If data formats are incompatible, they either give up or write Perl scripts to parse and transform the data format. This is hardly scalable when they need to process information about hundreds of genes in the shortest time period, or when they want to run large experiments with a large number of data sources and data items involved.

In the context of eScience, as well as in other types of commercial contexts, computer scientists have investigated different approaches for data integration and reconciliation. The traditional data warehousing approach harvests everything into a central database based on a unified data model, but data warehouses are expensive to maintain because of the ongoing “churn” of the underlying data sources and their data models [37]. XML-based Web Services have been employed to bring interoperable programmatic accesses to the underlying resources and workflow platforms integrate data by orchestrating services. Although Web Services bring interoperability to the data access, the inputs and outputs of different services are often incompatible, and hence still highly heterogeneous in terms of content. Instead of writing Perl scripts, bioinformaticians build *shim* services [39] to bridge the gap between services. The interoperability between data is not solved yet.

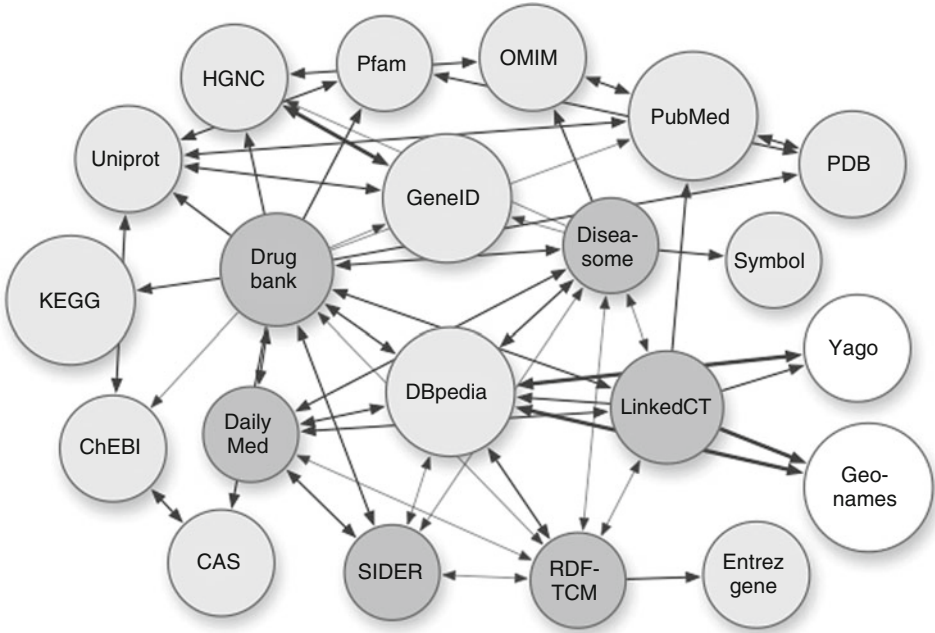
An approach to handle some of these problems has been proposed in the context of the Web of Data initiative, which aims at creating interoperability between data at a web-scale based on a stack of W3C standards and recommendations, including HTTP, and the use of URIs for identifying resources and RDF for describing them and the relationships among them. That is, in the Web of Data, resources are identified using URIs and described using RDF. A growing number of tools are capable of transforming legacy data,

such as XML databases, flat files, or relational databases, into the RDF format. A good number of them were described as a result of the W3C RDB2RDF Incubator Group (<http://www.w3.org/2005/Incubator/rdb2rdf/>) [63]. Some of these tools are D2R Server [40], R2O [41], or Triplify [42], to name but a few, and their main objective is to make relational databases accessible as RDF with or without transforming the source databases in the translation process. As a result of this transformation, some resources that are generated are identified by means of URIs, which should then be made dereferenceable so that users can put an URI in any Web browser and retrieve descriptions about this URI published on the Web, using HTTP.

Thus, scientists can browse the Web of Data, generated by means of this transformation process applied to existing data sources, as one big data warehouse hosted on the Web. For this, they can use specialized linked data browsers such as Tabulator [43], or hybrid browsers or plug-ins, such as the OpenLink Data Explorer plug-in of the Firefox Web browser (<http://ode.openlinksw.com/>). Alternatively, these RDF data can be made accessible through SPARQL endpoints, permitting application builders to programmatically query and retrieve information represented uniformly as RDF format, using standard SPARQL query languages. Thus, the Web of Data provides a higher departure point for building data integration applications.

There are several good examples of linked data sources in the scientific domain. One of them is the Linking Open Drug Data (LODD; <http://esw.w3.org/topic/HCLSIG/LODD>), which is a pilot study that facilitates the integration of drug-related data by publishing them in the Web of Data and investigates use cases to demonstrate how life science researchers as well as physicians and patients can benefit from this Web of Data (► *Fig. 17.5*). LODD is a task force of the W3C Health Care and Life Science Interest Group. It has published six databases covering information about drugs, traditional Chinese medicine, clinical trials, and diseases, see ► *Fig. 17.5*. The source databases are available either as XML or flat files. Existing tools such as D2R and IBM DB2 have been used to expose these legacy data as RDF [44]. Each data source is hosted in a separate store and accessible through a separate SPARQL endpoint. Links to external data sources, such as DBpedia or Entrez Gene database, or between these databases, are created using novel link creation software tools such as Silk [45] and LinQuer [46], which are capable of automatically creating links between data at a large scale. Overall, the published datasets contain more than 8.4 million RDF triples and 388,000 links to external data sources.

The current driving use case is to connect the knowledge from alternative medicine research with that from Western biomedical research. Lightweight Ajax (asynchronous JavaScript and XML) applications implementing distributed SPARQL queries are built allowing users to query the linked drug data. Patients can have one-stop-shop to search for alternative herbs that may help treat certain diseases as well as additional information about these herbs, including clinical trial information about the herb, active ingredients in the herb, and side effects reported for the ingredients. The application can also support biomedical researchers to investigate target genes of a herb for a specific disease. The application is currently being extended to find information about target genes and protein of herbs for particular diseases. This has the potential to reveal interesting opportunities



■ Fig. 17.5

A diagram of linked open drug data [44], courtesy of Anja Jentzsch. The dark gray represents data from the LODD group; the light gray represents other life science-related datasets; and the whites indicate data from other domains

for pharmaceutical companies to pursue. Researchers who have historically focused on Western medicines are becoming increasingly interested in exploring whether alternative medicines that have been used for thousands of years are working against the same targets.

Since the datasets are subject to change, the Web of Data as well as the links between data must be updated accordingly. Reloading an RDF graph is the easiest way to update the Web of Data. Tools for maintaining the links [47] on the Web of Data are also being evaluated and experimented with.

It is important to note that this is not the only effort for making scientific data accessible on the Web of Data. Other similar efforts are Bio2RDF, GeoSpecies, Gene Ontology annotations, Uniprot, etc. An updated list of the datasets that are available in the Linked Open Data cloud (approximately half of which is scientific oriented) can be found in the Linked Open Data website (<http://linkeddata.org/data-sets>).

While data cross-linking using the linked data approach is an important part of current research on eScience (and on semantic technologies in general), together with its publication and efficient exploitation by means of SPARQL queries, there is also an important part of scientific data that can be cross-linked using a combination of semantic technologies and user-interface technologies. A good example is the one provided in the context of the SemsorGrid4Env project (<http://www.semsorgrid4env.eu/>), which will be described next and which addresses a variety of data sources ranging from traditional relational databases

to data streams generated from sensor networks, in the context of critical decision taking in environmental science areas, such as flood modeling or fire detection.

A number of environmental management decisions depend on information provided by a variety of sources (e.g., legacy databases with historical data, real-time data coming from various kinds of sensor networks, mathematical models, and simulations). As the cost of deploying intelligent sensor networks falls, more sensor networks are being deployed and there is an increasing dependence on information coming from sensor networks for better situation assessment and decision-making. Some of the challenges that arise in this context are the following:

- Decision-making systems need to perform computations (for data fusion and integration, mining, and other purposes) on large amounts of real-time data stemming from heterogeneous, autonomously developed, and deployed sensor networks possibly combined with other existing data sources.
- They need to respond to sensed data in real-time, possibly adapting the behavior of the sensor network, to respond to emergency situations.
- They have to deal with very dynamic sensor network sources and changing application requirements that might call for using data in new, and possibly unexpected, ways outside the immediate scope of the project where they were deployed.

A set of commonalities can be found in all these contexts, namely, the fact that third parties integrate and enrich data from historical databases, live sensor networks that were deployed independently by different providers, and other sources in order to support early warning for fires or floods. In both cases, there is a clear need for third parties to be able to use data unexpectedly in a manner that was not previously envisioned (e.g., satellite products and data from tidal gauges), and make relevant data and information easily publishable using Web and mobile communications.

This project has investigated and built technological infrastructure for the rapid prototyping and development of open, large-scale semantic sensor webs for environmental management, particularly enabling:

- A semantically consistent view of several heterogeneous sensor networks as a global data resource
- Rapid development of services that combine real-world real-time data, coming from autonomous, heterogeneous sensor networks, with legacy historical data
- Rapid development of open, flexible, contextual knowledge-based thin applications (e.g., mash-ups) for environmental management

For this, a number of ontology networks have been used in the contexts of the application domains of the projects (floods and fire), which use existing ontologies in these areas, many of them coming from the SWEET suite of ontologies (<http://sweet.jpl.nasa.gov/ontology/>). Data streams coming from sensor networks, together with legacy historical data, are annotated according to these ontologies in order to produce RDF data streams and RDF static data, respectively, and these RDFized data sources can then be queried using spatiotemporal and stream-based extensions of the SPARQL query

language. The project also proposes a new means to describe linked data for RDF data streams that follows the same principles as that of the linked data initiative, although considering the differences that data streams bring in. Finally, a registry of available data sources, described according to a core set of ontologies based on the standard Open Geospatial Consortium (OGC) Observation and Measurement model, is generated and can be queried using the query language extensions identified above, so as to allow for the discovery of existing sensor networks or, in general, environmental data sources in a specific spatial region and in a window of time. All these data sources can then be mashed up by users in unexpected ways, by means of the appropriate user interfaces and mash-up management infrastructure. More information on the Web of Data can be found in [▶ Semantic Annotation and Retrieval: Web of Data](#).

17.4.6 Stopping the Division Between Data, Experiments, and Scientific Publishing: Prospect and Semantic Publications

While the value of making data open and accessible is widely recognized, as described in [▶ Sect. 17.4.5](#), data sharing remains a sensitive topic for scientists because in most cases they must protect their research results before they are published, given the fact that scientific publication is still the main hallmark of their academic achievement [48]. Digital publishing has the potential to create a high-quality, citable research data space, at the same time promoting scholar journal publication.

While the number of research articles grows at an exponential pace, the amount of data made accessible along with these articles is much smaller, since most of the data described in those articles as original data, results, or test beds are normally described only in paper, but not provided together with the research articles. However, scientists are eager to have the data within the article accessible in an actionable form [49], for a variety of reasons: they can perform similar experiments on similar objects and compare their results with those of other competing systems (as it has been the case traditionally in a good number of competitions, such as Text Retrieval Conference (TREC)), they can assess better the validity of the results presented in a scientific publication, they can perform mega-research by mashing up results from experiments on similar objects, performed under different conditions, by different research groups, or at different time and location, etc. The current division between data, experiments, and publications, together with the strong focus on paper-based publications rather than data and experiments, prevents the economy of reusing and recycling *past* science from being fully exploited. This section shows two example application scenarios to demonstrate the added value to scientists' research by making research data accessible in digital publications.

The project prospect (<http://www.prospectproject.org>) of the Royal Society of Chemistry (RSC) Publishing demonstrates the prospect of building an open chemistry data Web. RSC Publishing is a scientific publisher in chemical science that publishes

25 journals and 8,000 articles every year. It covers a broad spectrum of chemical sciences, from system biology to physical and theoretical chemistry. One of the activities that this project focuses on is in the use of the open-source text mining software Open Source Chemical Analysis Routines (OSCAR) (<http://sourceforge.net/projects/oscar3-chem/>) [50] to extract both chemical information and terms from research articles and in their annotation with unique identifiers and with terms from ontologies:

- The extracted chemical compounds are identified using IUPAC international chemical identifier (InChI) numbers, which act as a kind of URI. These compounds are described using RDF; and are linked to public databases (such as PubChem) through the unique InChI identifiers.
- A suite of ontologies are used to mark up extracted terms from the article: chemical entities of biological interest (ChEBI) [51] are used to annotate chemical classes, parts, and nanoparticles; the name reaction ontology (RXNO) (<http://www.rsc.org/ontologies/RXNO/>) to annotate the name reactions; the Gene Ontology (GO) and the Sequence Ontology (SO) to describe genomic data such as gene functions and nucleotide and polypeptide sequences; and the Cell Ontology (CL) to define the cell types.

The semantically enriched articles highlight the chemical compounds within, and provide additional information about a compound, such as its Simplified Molecular Input Line Entry Specification (SMILES) string [52], which gives a compact, computerized interpretation about its chemical structure, and links to related RSC articles (see an example at <http://dx.doi.org/10.1039/b613656g>). Scientists can also browse articles using the ontological terms and search for related articles by the hierarchical classification of these terms. This mixture of information across different disciplines enhances interdisciplinary research and reduces scientists' burden of keeping up with literature published by different communities. The knowledge structure presented by ontologies also lowers the barriers for users not specialized in a particular area to easily navigate scientific literature [53].

Another example is the one provided in [54], where the authors create an exemplar enhancement to a recent biomedical research article (<http://dx.doi.org/10.1371/journal.pntd.0000228>) studying infectious disease, published by the innovation leading scientific publisher PLoS. This enhancement (available at <http://dx.doi.org/10.1371/journal.pntd.0000228.x001>) reflects their definition of *semantic publication*, “as anything that enhances the meaning of a published journal article, facilitates its automated discovery, enables its linking to semantically related articles, provides access to data within the article in actionable form, or facilitates integration of data between papers” [49]. The goal of this exercise is to demonstrate that the use of very simple Web technologies (such as CSS, HTML, and hyperlinks), which can be easily mastered by scientists and adopted by publishers, can lead to a large improvement on the usefulness of online research articles.

After communication with the authors of the article, research data within the article are downloaded as Excel spreadsheets, a format commonly used by scientists to process and share their research data. These data are uniquely identified using digital object

identifiers (DOIs) (<http://www.doi.org>) by collaborating with PLoS. This is essential for making research data citable.

With the raw research data being available, the distribution of the antibodies can now be overlaid on top of the Google Maps to provide readers a better idea about the geo-location and geographical features of the study area. The value of making research data accessible is also demonstrated by correlating them with data from previous studies to, for example, confirm the studied area as having a high-incidence of leptospirosis. The figures in the article are made actionable; readers can move and overlay map figures capturing different aspects of the study on top of each other to draw new hypotheses [54].

In addition to identifying and annotating biological and disease-related terms with ontologies and linking them to third-party data sources, like the Universal Biological Indexer and Organizer (uBio, <http://www.ubio.org>), this exemplar also provides a “digital abstract” about the article, to facilitate cross-links between related articles and discovery of the article. The abstract summarizes the topic studied in the article, the hypothesis, the indicator of infection, the assay used to detect it, etc. It is represented in RDF format, defined using the citation typing ontology (<http://purl.org/net/cito>) [55], and these machine-processable metadata are embedded in the HTML Web page as RDFa.

Enhancement of scholarly publishing has yet to reach production level. The total number of enhanced articles from RSC Publishing is less than 10% of its yearly publications, and the semantic enhancement by Shotton et al. largely relies on manual processing. The winner of the recent Elsevier Grand Challenge (<http://www.elseviergrandchallenge.com/>) excels at accurate knowledge extraction from text in natural language [56]. However, the revolution of digital publishing cannot be achieved without the participation of scientists themselves, who have the authority and expertise to verify the semantic annotations provided by machines and own the research data to be made accessible. Scientists will share their research data if they are enforced by pioneering journal publishers or motivated by the possibility of increasing the citation of their research papers. If research data themselves became citable, a new reward scheme could be created based on the citation index of research data. To increase the visibility of data they must be supplied with structured metadata for discovery and cross-link, as well as with provenance information, for trust and credibility. One way to make data citable is using unique identifiers, such as InChI or DOI. However, InChI cannot represent all kinds of molecules; and the central administrations of DOIs, the International DOI foundation (<http://www.doi.org>) and CrossRef (<http://www.crossref.org>), are yet ready to publish DOIs for data.

17.5 Related Resources

After the presentation of different types of projects and initiatives in the areas identified as main challenges for the combination of eScience and semantic technologies, this section aims at providing an additional non-exhaustive list of reference resources that are related to the development of semantic eScience and have not been covered in detail in this

chapter. The aim is to provide pointers to other interesting works that share the vision of semantic eScience, so that readers can find potential leads to go into more depth in this area.

These resources are classified in the same way as we have done for the exemplar applications.

17.5.1 Management of Large Volumes of Research Data

There are several systems that provide support for the search and discovery of RDF data on the Web, pointing to large volumes of research data. Some of the most relevant are the following:

- Swoogle (<http://swoogle.umbc.edu/>) [57] was one of the first Google-like search engines for the Semantic Web, providing support for searching specific terms that may appear in RDF data sources or ontologies, or complete ontologies.
- Similarly, the Watson search engine (<http://watson.kmi.open.ac.uk/>) [58] can be also used for searching RDF data and ontologies across the Web, with a special focus on ontologies. More information on Swoogle and Watson can be found in [Semantic Web Search Engines](#).
- Other RDF aggregation systems that can be used for search are SWSE (<http://swse.deri.org/>) [59], which is a system that crawls RDF triples across the Web of Data and stores them, including their provenance, in a global database, or Sindice (<http://sindice.com/>) [60], which is similar in purpose, and has been developed in the same group. The latter not only crawls and indexes RDF triples but also the linked data on the Web and SPARQL endpoints. Its indexer is based on semantic sitemap (<http://sw.deri.org/2007/07/sitemapextension/>), which enables it to continuously update its indexing with reduced cost. Both systems have been the basis for the generation of the search and publishing system SIG.MA (<http://sig.ma/>), Semantic Information MASHup, which provides a holistic approach for automatic semi structured data discovery and consolidation, and is consolidating as one of the main general-purpose entry points for semantic search.

17.5.2 Support for Multidisciplinary Research

There are a good number of ontology repositories that aim at providing support for ontology search and selection in specific communities or domains, playing the key role for facilitating the interpretation of data from different sub-domains. Some of these repositories are the following:

- The NCBO BioPortal (<http://bioportal.bioontology.org/>) and the Open Biomedical Ontologies (<http://www.obofoundry.org/>), which contain a collection of ontologies that are being actively used in the biomedical communities

- The Marine Metadata Repository (<http://marinemetadata.org>), which contains a set of ontologies that are commonly used across a wide range of environmental science communities

Besides, there are other tools that provide support, by means of sets of generic or specialized components, to different activities related to eScience, such as text mining, specialized search on collection of resources, metadata storage, and querying. An example of such a tool is the Adaptive Information Disclosure Application (AIDA) search and annotation tool (<http://www.adaptivedisclosure.org/aida>). This tool allows scientists to collaborate remotely, annotating, interpreting, and searching large collection of documents in a heterogeneous format that are stored in distributed locations.

17.5.3 Knowledge Reuse and Reinterpretation

One of the most important activities in terms of allowing the description and exchange of ROs is the OAI-ORE (<http://www.openarchives.org/ore/>), which has been already mentioned in the section that covered the myExperiment platform. OAI-ORE provides a standard for describing and exchanging aggregations of Web resources and ROs. These resources can be any digital resource object accessible on the Web and represented in any format, including text, images, data, and video. Such a standard will play an important role for semantic publishing, allowing ROs shared by the scientists at each step of their research life cycle to be aggregated to form new knowledge and to permit mega-research.

17.5.4 Search and Discover the Right Data and Tools

The following are examples of specialized frameworks and tools that are being used in the context of semantic eScience applications. Again, we only mention a few of them, from the large number of frameworks and tools available, with the aim of providing only some links to interesting activities that can lead readers to other activities not covered in detail in this chapter. We start with general frameworks for search and querying (besides those presented at the beginning of this section)

- Semantic Automated Discovery and Integration (SADI) is a framework that uses the SPARQL query language to search in the deep Web. This framework has been applied, for instance, to the CardioSHARE project (<http://cardioshare.biordf.net/cardioSHARE/query>). In SADI, a SPARQL query posed by a user or a system to the SADI framework will be translated into a search for Web Services that match the predicates given in the input SPARQL query. Then the query results from the identified and selected Web Services will be mashed together and returned to the user, as if they were coming from any other type of more traditional data source. Instead of

workflows, SADI allows users to perform data integration and query federation through a single SPARQL query.

- The AIDA search interface (<http://www.adaptivedisclosure.org/aida/docs/search/search-demos>) allows distributed users to perform free-text search, SKOS-like search to multiple RDF stores, or SPARQL endpoints that are indexed by their extended Lucene indexing and search engine.
- The Query Federation case study by the HCLS BioRDF task force [61] reviews the latest development in query federation in the background of improved RDF triple technologies and the emergence of SPARQL standards and endpoints, linked data, and the vocabulary of interlinked datasets.

A common means to collect useful resources that can be later searched, discovered, and reused is the use of wiki-style websites for sharing research data. Some representative examples of such websites are the following:

- EcoliWiki (<http://ecoliwiki.net>), which integrates information from 19 websites relevant to *E. coli*
- PDBWiki (<http://pdbwiki.org>), which is a community-annotated knowledge base of biological molecular structures from the Protein Data Bank (PDB)
- Proteopedia (<http://www.proteopedia.org>), which is an interactive encyclopedia of 3D structures of proteins, RNA, DNA, and other molecules
- WikiGenes (<http://www.wikigenes.org>), which contains information about thousands of genes, chemicals, pathologies, etc.
- WikiPathways (<http://www.wikipathways.org>), which is focused on the curation of biological pathways by and for the scientific community
- OpenWetWare (<http://openwetware.org/>), which shares tools and protocols for scientific research, aiming at promoting the sharing of information, know-how, and wisdom among researchers and groups from the biology and biological engineering domains

17.5.5 Cross-Linking of Data

Some of the most relevant examples in this direction were already identified in the corresponding section. Other relevant examples are the following:

- The Bio2RDF project (<http://bio2rdf.org/>) [62] aims at providing an atlas of postgenomic knowledge.
- OpenCalais (<http://www.opencalais.com/>) from Thomson Reuters allows the extraction of information from unstructured documents and connects information together with the Web of Data.
- Microsoft Amalga (<http://www.microsoft.com/amalga/products/microsoftamalgauis/>) allows the integration of health care data to support connected health.

17.5.6 Stopping the Division Between Data, Experiments, and Scientific Publishing

Besides the examples provided in the corresponding section, one additional example of how to combine publications and data is Concept Web (<http://www.wikiprofessional.org>), which enables the exploration of factual and associative relationships between concepts, visualizing the so-called Knowlets, which are a novel way of showing how concepts relate, and viewing the source publications that serve as Knowlet building blocks.

17.6 Conclusions and Future Issues

This chapter has looked at a number of pioneering semantic eScience projects that cover a diversity of application domains including bioinformatics, biology, environmental science, Earth observation, and chemistry. They reflect how the state-of-the-art semantic eScience can support data-intensive science, facilitate experimental knowledge reuse and recycling among scientists, lower the barriers of knowledge exchange for interdisciplinary research, and bridge the gap between data from different sources and the gap between data sharing and digital scholarly publication.

The application of semantic technologies to eScience is driven by the common need for coping with the heterogeneity of distributed computing. The subtle difference of terminologies from different sub-domains must be well defined and made explicit; information from different sources represented in different formats or languages must be made interoperable for scientific collaboration. The example projects from this chapter have demonstrated how semantic metadata make it possible to enable scientists to exploit the large volume of research outcomes, for example, by retrieving related articles based on the semantic annotations of their content or by discovering services best fitting their needs without knowing where the services are or what they can do. Semantic metadata also make it easier for scientists to ask questions and to understand the answers without having to adequately understand every discipline.

Although the maturing of semantic technologies means better support for semantic eScience applications, most of the example projects presented here are not yet at the production stage. The following issues are reflected and should be addressed in the future in order to improve the uptake of semantic technologies in eScience projects and environments:

The usability of semantic technologies: The success of technology innovation fundamentally lies in the usability of user interfaces and the “fitting in” of technologies to existing working practices (although as we have mentioned in [Sect. 17.1](#), in some cases technologies can also change the way that working practices are done, generating new ones). Scientists expect an intuitive presentation of the semantic enhancement to publications and ROs; the last thing they expect is to learn complex text-mining tools as part of the process of literature review, or complex query languages in order to find or understand the data resources that they are discovering. Although new technologies could introduce

new, more productive methods to the users, the changes should be in general introduced incrementally; scientists would not give up their existing methods unless they were convinced by the benefits.

Heterogeneous identities make cross-links hard: The key to cross-linking distributed statements about the same data item is through a unique identity of the item. However, the heterogeneity of data identities is an historical issue; each data publisher provides a different way to identify an item and deciding whether two gene records are the same often leads to philosophical discussions. A consensus of identities cannot be reached without a community-wide agreement. Efforts such as Shared Names (<http://sharedname.org/>) and the Concept Web Alliance (<http://conceptweblog.wordpress.com/>), among others, bring together domain scientists, researchers from semantic eScience, and social scientists to address such issues.

Better support for search in the deep Web (of Data): Although semantic search has the potential to review implicit knowledge from the Web of Data, which contains a large volume of structured metadata about a wide spectrum of resources, the power of semantic search is yet to be comparable with Web search engines. The best solutions using semantic search are found in customized applications, which describe resources using ontologies from their specific domains and create pre-canned queries based on specific user requirements. Supporting larger-scale data mining across the Web of Data means that data must have unique identities and cross-links between distributed data sources must be accurately established so that the aggregation of knowledge and search across these data can be efficiently achieved as if they are from one web. Better algorithms are expected out of the thriving semantic search community. The performance and reliability of the state-of-the-art semantic technologies on managing and querying data at a web-scale are yet to be evaluated.

Trust and quality: Quality assessment of data should be a paramount task in order to ensure that the most appropriate and trustworthy data are made available and delivered to users. Scientific applications built upon the Web of Data will be of little value if scientists are skeptical of the quality of data. However, the criteria for evaluating the quality and trustworthiness of data are often subjective. Metadata such as provenance should be used for the assessment and the process must be automated as much as possible.

Knowledge transfer to get scientists involved: Innovation must be brought to the scientists who eventually revolutionize scientific research. It is scientists themselves who need to learn how to organize and index their fast-growing data, who have the knowledge to share with the science community, and who own the research data that should be made publicly accessible on the Web or through research articles. The potential of knowledge recycling and the mega-research promised by computing technologies cannot be realized until these technologists have truly become part of the research methods and practices.

Knowledge transfer and education: The research outcomes from semantic eScience should be passed not only to engineering students who will continue to drive future innovations but also to students from other domains who will take the lead in the future art and science studies. Transferring knowledge to this audience will accelerate the involvement of the scientists and the cultural shift to boost knowledge sharing and adoption of cutting-edge technologies.

17.7 Cross-References

- Future Trends
- Querying the Semantic Web: SPARQL
- Semantic Annotation and Retrieval: RDF
- Semantic Annotation and Retrieval: Web of Data
- Semantic Annotations and Retrieval: Manual, Semiautomatic, and Automatic Generation
- Social Semantic Web
- Storing the Semantic Web: Repositories

References

1. Galperin, M., Cochrane, G.: Nucleic acids research annual database issue and the NAR online molecular biology database collection in 2009. *Nucleic Acids Res.* **37**(Database issue), 1–4 (2009)
2. Clery, D., Voss, D.: All for one and one for all. *Science* **308**(5723), 809 (2005)
3. McGuinness, D., Fox, P., Brodaric, B., Kendall, E.: The emerging field of semantic scientific knowledge integration. *IEEE Intell. Syst.* **24**(1), 25–26 (2009)
4. McGuinness, D., Fox, P., Cinquini, L., West, P., Garcia, J., Benedict, J., Middleton, D.: The virtual solar-terrestrial observatory: a deployed semantic web application case study for scientific research. In: *Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI-07)*, Vancouver, pp. 1730–1737 (2007)
5. Ludascher, B., Goble, C.: Guest editors' introduction to the special section on scientific workflows. *SIGMOD Rec.* **34**(3), 2 (2005)
6. Web Services activity statement: World Wide Web Consortium (W3C) activity statement. <http://www.w3.org/2002/ws/Activity.html> (Apr 2004). Accessed Dec 2010
7. Moreau, L., Ludascher, B., Altintas, I., Barga, R., Bowers, S., Callahan, S., Chin, J.R., Clifford, B., Cohen, S., Cohen-Boulakia, S.: Special issue: the first provenance challenge. *Concurr. Comput. Pract. Exp.* **20**(5), 409–418 (2008)
8. W3C semantic web activity. <http://www.w3.org/2001/sw/> (2001). Accessed 14 July 2009
9. Klyne, G., Carroll, J.J., McBride, B.: Resource Description Framework (RDF): concepts and abstract syntax, W3C recommendation. www.w3.org/TR/rdf-concepts/ (2004). Accessed Dec 2010
10. Gruber, T.: A translation approach to portable ontology specifications. *Knowl. Acquis.* **5**, 199–220 (1993)
11. Brickley, D., Guha, R.: RDF vocabulary description language 1.0: RDF schema, W3C recommendation (2004)
12. McGuinness, D., Van Harmelen, F.: OWL web ontology language overview, W3C recommendation (2004)
13. Brickley, D., Miller, L.: FOAF vocabulary specification 0.9. Namespace document (2007)
14. Breslin, J., Harth, A., Bojars, U., Decker, S.: Towards semantically-interlinked online communities. In: *Proceedings of the Second European Semantic Web Conference (ESWC 2005)*, Heraklion. *Lecture Notes in Computer Science*, vol. 3532, pp. 500–514. Springer, Berlin (2005)
15. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF, W3C Recommendation. <http://www.w3.org/TR/rdf-sparql-query/> (2008). Accessed 15 Jan 2008
16. Clark, K.G., Feigenbaum, L., Torres, E.: SPARQL protocol for RDE, W3C Recommendation. <http://www.w3.org/TR/rdf-sparql-protocol/> (2008). Accessed 15 Jan 2008
17. Berners-Lee, T., Cailliau, R.: The world-wide web. *Commun. ACM* **37**(8), 76–82 (1992)
18. Berners-Lee, T.: Linked data-design issues. <http://www.w3.org/DesignIssues/LinkedData.html> (2006). Accessed 14 July 2009
19. Bizer, C., Health, T., Berners-Lee, T.: Linked Data – the story so far. *Int. J. Semant. Web Inf. Syst.* **5**(3), 1–22 (2009). (Special Issue on Linked Data)

20. Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Berners-Lee, T.: RFC2616: hypertext transfer protocol – HTTP/1.1. Internet RFCs (1999). Accessed Dec 2010
21. Berners-Lee, T., Fielding, R., Masinter, L.: RFC 2396: Uniform Resource Identifiers (URI): generic syntax. Internet RFCs (1998)
22. Fox, P., McGuinness, D., Cinquini, L., West, P., Garcia, J., Benedict, J., Middleton, D.: Ontology-supported scientific data frameworks: the virtual solar-terrestrial observatory experience. *Comput. Geosci.* **35**(4), 724–738 (2009)
23. Fox, P., Cinquini, L., McGuinness, D., West, P., Garcia, J., Benedict, J., Zednik, S.: Semantic web services for interdisciplinary scientific data query and retrieval. In: Proceedings of the AAAI Semantic e-Science Workshop, Vancouver, pp. 42–50 (2007)
24. Wright, R., Sánchez-Gestido, M., Gómez-Pérez, A., Pérez-Hernández, M.S., González-Cabero, R., Corcho, O.: A semantic data grid for satellite mission quality analysis. In: Proceedings of the Seventh International Semantic Web Conference (ISWC 2008), Karlsruhe. Lecture Notes in Computer Science, vol. 5318, pp. 818–832. Springer, Heidelberg (2008)
25. ESA bulletin number 106, “EnviSat special issue”. <http://www.esa.int/esapub/pi/bulletinPI.htm>. Accessed Dec 2010
26. De Roure, D., Goble, C., Stevens, R.: Designing the myexperiment virtual research environment for the social sharing of workflows. In: Proceedings of the Third IEEE International Conference on e-Science and Grid Computing, Bangalore, pp. 603–610 (2007)
27. De Roure, D., Goble, C., Alekseyevs, S., Bechhofer, S., Bhagat, J., Cruickshank, D., Fisher, P., Hull, D., Michaelides, D., Newman, D.: Towards open science: the myExperiment approach. *Concurr. Comput. Pract. Exp.* **22**(17), 2335–2353 (2010)
28. De Roure, D., Goble, C., Bhagat, J., Cruickshank, D., Goderis, A., Michaelides, D., Newman, D.: myExperiment: defining the social virtual research environment. In: Proceedings of the Fourth IEEE International Conference on e-Science, Indianapolis, pp. 182–189 (2008)
29. Newman, D., Bechhofer, S., De Roure, D.: myExperiment: an ontology for e-research. In: Workshop on Semantic Web Applications in Scientific Discourse, Washington, DC (2009)
30. Preece, A., Jin, B., Pignotti, E., Missier, P., Embury, S., Stead, D., Brown, A.: Managing information quality in e-Science using semantic web technology. In: Proceedings of the Third European Semantic Web Conference (ESWC 2006), Budva. Lecture Notes in Computer Science, vol. 4011, pp. 472–486. Springer, Berlin (2006)
31. Missier, P., Embury, S., Greenwood, M., Preece, A., Jin, B.: Quality views: capturing and exploiting the user perspective on data quality. In: Proceedings of the 32nd Very Large Data Bases (VLDB 2006), Seoul, pp. 977–988 (2006)
32. Missier, P., Embury, S., Greenwood, M., Preece, A., Jin, B.: Managing information quality in e-Science: the curator workbench. In: Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data, Beijing, pp. 1150–1152 (2007)
33. Belhajjame, K., Embury, S., Paton, N.: On characterising and identifying mismatches in scientific workflows. In: Proceedings of the Third International Workshop on Data Integration in the Life Sciences (DILS 2006), Hinxtion. Lecture Notes in Computer Science, vol. 4075, pp. 240–247. Springer, Berlin (2006)
34. Goble, C., Belhajjame, K., Tanoh, F., Bhagat, J., Wolstencroft, K., Stevens, R., Nzuobontane, E., McWilliam, H., Laurent, T., Lopez, R.: BioCatalogue: a curated web service registry for the life science community. In: Microsoft eScience Conference, Nature Precedings, Indianapolis, p. 3132 (2008)
35. Wolstencroft, K., Alper, P., Hull, D., Wroe, C., Lord, P.W., Stevens, R.D., Goble, C.A.: The myGrid ontology: bioinformatics service discovery. *Int. J. Bioinform. Res. Appl.* **3**(3), 303–325 (2007)
36. Belhajjame, K., Embury, S., Paton, N., Stevens, R., Goble, C.: Automatic annotation of web services based on workflow definitions. *ACM Trans. Web* **2**(2), 1–34 (2008)
37. Stein, L.: Integrating biological databases. *Nat. Rev. Genet.* **4**(5), 337–345 (2003)
38. Stein, L.: Towards a cyberinfrastructure for the biological sciences: progress, visions and challenges. *Nat. Rev. Genet.* **9**(9), 678–688 (2008)
39. Hull, D., Stevens, R., Lord, P., Goble, C.: Integrating bioinformatics resources using shims. Poster

- Abstract in Proceedings of the 12th International Conference on Intelligent Systems for Molecular Biology, Glasgow (2004)
40. Bizer, C., Cyganiak, R.: D2R server-publishing relational databases on the semantic web. Poster Session of the Fifth International Semantic Web Conference (ISWC 2006), Athens (2006)
 41. Barrasa, J., Corcho, O., Gómez-Pérez, A.: R2O, an extensible and semantically based database-to-ontology mapping language. In: Proceedings of the Second Workshop on Semantic Web and Databases (SWDB 2004), Toronto (2004)
 42. Auer, S., Dietzold, S., Lehmann, J., Hellmann, S., Aumueller, D.: Triplify: light-weight linked data publication from relational databases. In: Proceedings of the 18th International Conference on the World Wide Web (WWW 2009), Madrid, pp. 621–630 (2009)
 43. Berners-Lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J., Lerer, A., Sheets, D.: Tabulator: exploring and analyzing linked data on the semantic web. In: Proceedings of the Third International Semantic Web User Interaction Workshop (SWUT 2006), Athens, GA (2006)
 44. Jentzsch, A., Zhao, J., Hassanzadeh, O., Cheung, K.-H., Samwald, M., Andersson, B.: Linking open drug data. In: Triplification Challenge of I-Semantics'09, Graz (2009)
 45. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Silk – a link discovery framework for the web of data. In: Proceedings of the Linked Data on the Web Workshop at WWW 2009, Madrid (2009)
 46. Hassanzadeh, O.: A framework for semantic link discovery over relational data. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009), Hong Kong, pp. 1027–1036 (2009)
 47. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Discovering and maintaining links on the web of data. In: Proceedings of the Eighth International Semantic Web Conference (ISWC 2009), Chantilly. Lecture Notes in Computer Science, vol. 5823, pp. 650–665. Springer, Berlin (2009)
 48. Murray-Rust, P.: Chemistry for everyone. *Nature* **451**(7179), 648–651 (2008)
 49. Shotton, D.: Semantic publishing: the coming revolution in scientific journal publishing. *Learn. Publ.* **22**(2), 85–94 (2009)
 50. Corbett, P., Murray-Rust, P.: High-throughput identification of chemistry in life science texts. *Comput. Life Sci. II* **4216**, 107–118 (2006)
 51. Degtyarenko, K., Matos, P., Ennis, M., Hastings, J., Zbinden, M., McNaught, A., Alcantara, R., Darsow, M., Guedj, M., Ashburner, M.: ChEBI: a database and ontology for chemical entities of biological interest. *Nucleic Acids Res.* **36**, D344–D350 (2007)
 52. Weininger, D.: SMILES, a chemical language and information system. *J. Chem. Inf. Comput. Sci.* **28**(1), 31–36 (1988)
 53. Research Information: Semantic enrichment boosts information retrieval. http://www.researchinformation.info/features/feature.php?feature_id=127 (2007). Accessed Dec 2010
 54. Shotton, D., Portwin, K., Klyne, G., Miles, A.: Adventures in semantic publishing: exemplar semantic enhancements of a research article. *PLoS Comput. Biol.* **5**(4), e1000361 (2009)
 55. Shotton, D.: CiTO, the Citation Typing Ontology, and its use for annotation of reference lists and visualization of citation networks. In: Proceedings of the 12th Annual Bio-Ontologies Meeting, Stockholm (2009)
 56. Pafilis, E., O'Donoghue, S., Jensen, L., Horn, H., Kuhn, M., Brown, N., Schneider, R.: Reflect: augmented browsing for the life scientist. *Nat. Biotechnol.* **27**(6), 508–510 (2009)
 57. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R., Peng, Y., Reddivari, P., Doshi, V., Sachs, J.: Swoogle: a search and metadata engine for the semantic web. In: Proceedings of the 13th ACM International Conference on Information and Knowledge Management (CIKM 2004), Washington, DC, pp. 652–659 (2004)
 58. d'Aquin, M., Sabou, M., Motta, E.: Reusing knowledge from the semantic web with the Watson Plugin. In: Demo, Proceedings of the Seventh International Semantic Web Conference (ISWC 2008), Karlsruhe. Lecture Notes in Computer Science, vol. 5318. Springer, Berlin (2008)
 59. Harth, A., Umbrich, J., Decker, S.: Multicrawler: A pipelined architecture for crawling and indexing semantic web data. In: Proceedings of the International Semantic Web Conference (ISWC 2006), Sardinia. Lecture Notes in Computer Science, vol. 3729, pp. 258–271. Springer, Heidelberg (2006)

60. Tummarello, G., Delbru, R., Oren, E.: Sindice.com: weaving the open linked data. In: Proceedings of the Sixth International Semantic Web Conference (ISWC 2007), Busan. Lecture Notes in Computer Science, vol. 4825, pp. 552–565. Springer, Heidelberg (2007)
61. Cheung, K., Frost, H., Marshall, M., Prud'hommeaux, E., Samwald, M., Zhao, J., Paschke, A.: A journey to semantic web query federation in life sciences. *BMC Bioinform.* **10**(Suppl 10), S10 (2009)
62. Belleau, F., Nolin, M., Tourigny, N., Rigault, P., Morissette, J.: Bio2RDF: towards a mashup to build bioinformatics knowledge systems. *J. Biomed. Inform.* **41**(5), 706–716 (2008)
63. Sahoo, S.S., Halb, W., Hellmann, S., Idehen, K., Thibodeau Jr, T., Auer, S., Sequeda, J., Ezzat, A.: A survey of current approaches for mapping of relational databases to RDF. W3C Technical Report. http://www.w3.org/2005/Incubator/rdb2rdf/RDB2RDF_SurveyReport.pdf (2009)

18 Knowledge Management in Large Organizations

John Davies¹ · Paul Warren²

¹British Telecommunications Plc, Ipswich, UK

²Eurescom GmbH, Heidelberg, Germany

18.1	<i>Scientific and Technical Overview</i>	739
18.1.1	Introduction	739
18.1.2	The Challenges for Organizational Knowledge Management	740
18.1.3	Finding Information and Organizing Information so that It Can Be Found	741
18.1.3.1	Defects of the Conventional Search Engine	741
18.1.3.2	Semantic Indexing and Retrieval	743
18.1.3.3	Storing Information for Easier Retrieval	744
18.1.4	Sharing Knowledge Across the Organization	745
18.1.5	Helping with Processes	746
18.1.6	Information Integration	746
18.1.6.1	The Challenges of Information Integration	746
18.1.6.2	Approaches to Information Integration in the Enterprise	747
18.1.6.3	Using Ontologies for Information Integration	749
18.1.6.4	Research Themes in Information Integration	750
18.1.7	Integrating Structured and Unstructured Information	751
18.1.7.1	The Need to Analyze Text	751
18.1.7.2	Combining the Statistical and Linguistic Approaches	754
18.1.8	Sharing Knowledge Between Organizations	756
18.1.9	Another Look at Ontologies	757
18.2	<i>Example Applications</i>	758
18.2.1	Semantic Search, Browse, and Information Storage	759
18.2.1.1	Squirrel: An Example of Semantic Search and Browse	759
18.2.1.2	SEKTagent: A Different View on Semantic Search	762
18.2.1.3	Semantic Filing: TagFS and SemFS	763
18.2.1.4	Commercial Activities	764
18.2.2	Semantic Information Sharing	765
18.2.2.1	Effective Document Sharing with Semantic Technologies	765
18.2.3	The Semantic Desktop: Supporting the User Throughout His Work	770
18.2.3.1	Sharing Information and Metadata Across Applications and Desktops ..	770
18.2.3.2	Understanding User Context	772

18.2.4	Graphical and Semiautomatic Approaches to Information Integration ...	773
18.2.5	Extracting and Exploiting Semantics from Unstructured Information ...	775
18.2.5.1	Software for Text Analytics	775
18.2.5.2	Extracting Information from the World Wide Web	776
18.2.6	Sharing Information Across Organizations	777
18.2.6.1	An Example from Medicine	777
18.2.6.2	The Web of Linked Data	778
18.3	<i>Related Resources</i>	779
18.3.1	Semantic Web Interest Group: Case Studies and Use Cases	780
18.4	<i>Future Issues</i>	781
18.4.1	Web2.0 and Ontologies	781
18.4.2	Integrating into and across Enterprises	781
18.5	<i>Cross-References</i>	782

Abstract: This chapter provides an overview of the knowledge management (KM) problems, and opportunities, faced by large organizations, and indeed also shared by some smaller organizations. The chapter shows how semantic technologies can make a contribution. It looks at the key application areas: finding and organizing information; sharing knowledge; supporting processes, in particular informal processes; information integration; extracting knowledge from unstructured information; and finally sharing and reusing knowledge across organizations. In each application area, the chapter describes some solutions, either currently available or being researched. This is done to provide examples of what is possible rather than to provide a comprehensive list. The chapter also describes some of the technologies which contribute to these solutions; for example, text mining for analyzing documents or text within documents; and natural language processing for analyzing language itself and, for example, identifying named entities. Most fundamentally, the use of ontologies as a form of knowledge representation underlies everything talked about in the chapter. Ontologies offer great expressive power; they provide enormous flexibility, with the ability to evolve dynamically unlike database schema; and they make possible machine reasoning. The chapter concludes by identifying the key trends and describing the key challenges to be faced in the development of more powerful tools to support knowledge work.

18.1 Scientific and Technical Overview

18.1.1 Introduction

This chapter is concerned with how semantic technologies can make a difference to managing knowledge in large organizations. That the management of knowledge in organizations is a problem, and also an opportunity, is of no doubt. The management scientist Peter Drucker has commented that “the most important contribution management needs to make in the 21st century is to . . . increase the productivity of knowledge work” [1]. He identified increased productivity of manual work as a major distinguishing feature of successful organizations in the twentieth century and saw increased productivity of knowledge work as a similarly distinguishing feature of successful organizations in the twenty-first century. To Drucker, knowledge work was work where “the task does not program the worker,” that is, where the worker himself or herself has to make choices about what he does. Writing at the very end of the twentieth century, he estimated knowledge workers, that is, those involved in this sort of work, as possibly already composing two fifths of the US workforce. Note that in all discussion about “knowledge work” and “knowledge workers” it is important not to assume too elitist a definition. The users of the technology described in this chapter are not limited to people who have a graduate-level education but include everyone who works with knowledge. Indeed, in the paper referenced, Drucker talks at length about what he calls “technologists,” that is, people who work with their hands and yet also perform knowledge work. As Drucker notes, these can range from surgeons to telephone repair technicians. As a management scientist, Drucker’s concern was with

management's contribution to increasing knowledge worker productivity. The related concern here is with technology's contribution.

Organizations most conscious of the importance of knowledge, and of managing knowledge, tend to be those in the business of selling knowledge, that is, consultancies. Such organizations usually invest a significant amount of money in KM technology and employ KM professionals to support the sharing and reuse of knowledge. However, all organizations experience problems in the managing of knowledge, and in general the larger the organization, the greater the problems experienced. Over the last few decades, a large amount of research has been undertaken into how to improve the management of knowledge. This research has been technological, organizational, and user-oriented. This handbook is primarily about technology and the focus in this chapter is on the application of semantic technology to KM. However, the authors of this chapter believe that the technological, organizational, and user aspects of KM cannot be seen in isolation, but rather that understanding their interaction is important to designing successful KM systems.

The chapter is entitled "Knowledge Management in Large Organization." In some places, it refers to "information management," in others "knowledge management" (KM). The former is a necessary precursor to the latter. A widely quoted articulation of the difference between information and knowledge is due to R. L. Ackoff [2]. Ackoff sees information as useful data, providing answers to the "who," "what," "where," and "when" questions. Knowledge, on the other hand, enables the application of information; it answers the "how" question. The chapter will adhere as far as possible to this distinction, although the choice of terminology will also be guided by what seems the more natural English usage in any given circumstance.

This chapter makes a number of references to research projects and also to commercial systems. In general, these are chosen as examples to illustrate possible approaches. This chapter is not an exhaustive review of such systems and the examples given are simply those known to the authors. Their inclusion here does not imply any particular merit over systems not described here.

18.1.2 The Challenges for Organizational Knowledge Management

For those concerned with the management of information and knowledge in an organization, there are a number of challenges:

- Enabling the user to find, or be proactively presented with, the right information to achieve a particular task. The information might be taken from a wide range of sources, including databases, an intranet or the Internet; or it might be an amalgam of information from various sources. Related to this is the need to organize information in a way in which it can be efficiently retrieved.
- Sharing knowledge across the organization. Here also, the knowledge may be in a database, intranet, or the Internet (explicit knowledge), or simply in an individual's head (tacit knowledge). The person who needs the knowledge, and the owner or creator of the knowledge, although colleagues, may even be located on different continents.

- Helping users to navigate the processes, often collaborative processes, of which their work is composed. Central to this is sharing metadata between applications, to support a particular goal. Also important is having an understanding of the user's current context, and what he or she is trying to achieve.
- The integration of associated information which is held in multiple databases across and outside the organization. Note that the concern here is specifically with information which is inherently structured.
- The integration of structured information held in corporate databases with unstructured information, for example, held on the corporate intranet. By merging information from all corporate sources, a complete picture of what the organization knows about a particular topic can be obtained.
- Organizations do not exist in isolation but collaborate commercially and for the purposes of research. That collaboration requires a sharing of information. Typically, different organizations will have different vocabularies for talking about their shared concerns. This creates an enlarged version of the enterprise database integration problem described above.

The importance of these challenges has been highlighted by an Economist Intelligence Unit report, which surveyed 565 executives from various industries [3] – 74% of respondents said “data gathering is a significant or very significant challenge” and 68% said the same about data-searching. In fact, 42% of the respondents could not find relevant information when needed, 58% rated the challenge of knowledge sharing and collaboration as 4 or 5 (on a scale of 1–5), and 52% similarly rated the challenge of data integration as 4 or 5. Further, bearing out the need for information integration, 54% said that “necessary information resides in silos.” Interestingly, users were more satisfied with the quality and quantity of information available than with the ease of access and ease of use of that information.

➤ [Sections 18.1.3–18.1.8](#) discuss these challenges in greater detail, explaining why systems which analyze information on the semantic level are important in solving these challenges; ➤ [Sect. 18.1.9](#) makes some remarks about the ontological approach to information management compared to that of relational databases. ➤ [Section 18.2](#) describes some applications of semantic technologies to the challenges previously discussed. ➤ [Section 18.3](#) lists some relevant resources. Finally, ➤ [Sect. 18.4](#) discusses future trends and unsolved challenges.

18.1.3 Finding Information and Organizing Information so that It Can Be Found

18.1.3.1 Defects of the Conventional Search Engine

The search engine has been the great success story of the World Wide Web. However, its use within the organization has been less successful and has created a degree of frustration.

An important reason for this is well known. The page rank algorithm, pioneered by Google, depends on the rich pattern of hyperlinks which exist on the Web but which are rarely to be found on the organizational intranet.

However, even at its most successful, the conventional search engine suffers from an approach based on text-string matching and consequent failure to interpret the semantics of a query or the semantics inherent in the documents being queried. In particular, the failure to identify polysemy; a similar failure to take account of synonymy and other forms of semantic connection between terms; an inability to make use of context; and less than optimal interpretation of results.

Polysemy

A difficulty with query terms is that they may have multiple meanings; this is called query term polysemy. As conventional search engines cannot interpret the sense of the user's search, the ambiguity of the query leads to the retrieval of irrelevant information.

Although the problems of query ambiguity can be overcome to some degree, for example, by careful choice of additional query terms, there is evidence to suggest that many people may not be prepared to do this. For example, an analysis of the transaction logs of the Excite WWW search engine [4] showed that Web search engine queries contain on average 2.2 terms. Comparable user behavior can also be observed on corporate intranets. An analysis of the queries submitted to BT's intranet search engine over a 4-month period between January 2004 and May 2004 showed that 99% of the submitted queries only contained a single phrase and that, on average, each phrase contained 1.82 keywords.

Synonymy and Semantic Links

Converse to the problem of polysemy is the fact that conventional search engines that match query terms against a keyword-based index will fail to match relevant information when the keywords used in the query are different from those used in the index, despite having the same meaning (index term synonymy). Although this problem can be overcome to some extent through thesaurus-based expansion of the query, the resultant increased level of document recall may result in the search engine returning too many results for the user to be able to process realistically.

In addition to an inability to handle synonymy and polysemy, conventional search engines are unaware of any other semantic links between concepts. Consider, for example, the following query:

“telecom company” Europe “John Smith” director

The user might require, for example, documents concerning a telecom company in Europe, a person called John Smith, and a board appointment. Note, however, that a document containing the following sentence would not be returned using conventional search techniques:

“At its meeting on the 10th of May, the board of London-based O2 appointed John Smith as CFO”

In order to be able to return this document, the search engine would need to be aware of the following semantic relations:

O2 is a mobile operator, which is a kind of telecom company.

London is located in the UK, which is a part of Europe.

A CFO is a kind of director.

Lack of Context

Many search engines fail to take into consideration aspects of the user's context to help disambiguate their queries. User context would include information such as a person's role, department, experience, interests, project work, etc. A simple search on BT's intranet demonstrates this. A person working in a particular BT line of business searching for information on their corporate clothing entitlement is presented with numerous irrelevant results if they simply enter the query "corporate clothing." More relevant results are only returned should the user modify their query to include further search terms to indicate the part of the business in which they work. As discussed above, users are in general unwilling to do this.

Presentation of Results

The results returned from a conventional search engine are usually presented to the user as a simple ranked list. The sheer number of results returned from a basic keyword search means that results navigation can be difficult and time consuming. Generally, the user has to make a decision on whether to view the target page based upon information contained in a brief result fragment. A survey of user behavior on BT's intranet suggests that most users will not view beyond the tenth result in a list of retrieved documents; only 17% of searches resulted in a user viewing more than the first page of results. Essentially, the requirement is to move from a document-centric view to a more knowledge-centric one (for example, by presenting the user with a digest of information gleaned from the most relevant results found as has been done in the Squirrel semantic search engine described later in this chapter).

18.1.3.2 Semantic Indexing and Retrieval

The previous section discussed the limitations of conventional textual search technology and indicated that these limitations were caused by a failure to interpret the semantics both in the query and in the textual corpus being interrogated. Chapter on [Semantic Annotations and Retrieval: Manual, Semiautomatic, and Automatic Generation](#) of this handbook has described techniques for the automatic creation of semantic annotations. As explained in [5], semantic indexing and retrieval can then be performed on top of the semantic annotations. Indexing can be done with respect to two semantic features: lexical concepts and named entities. In this way, a number of the problems discussed above can be overcome.

Lexical concepts are introduced to overcome the polysemy discussed earlier. Thus, a word with two different meanings will be associated with two different lexical concepts. Word-sense disambiguation techniques can be used to disassociate these meanings [6]. Similarly, knowing that two words or phrases are associated with the same lexical concept enables the system to

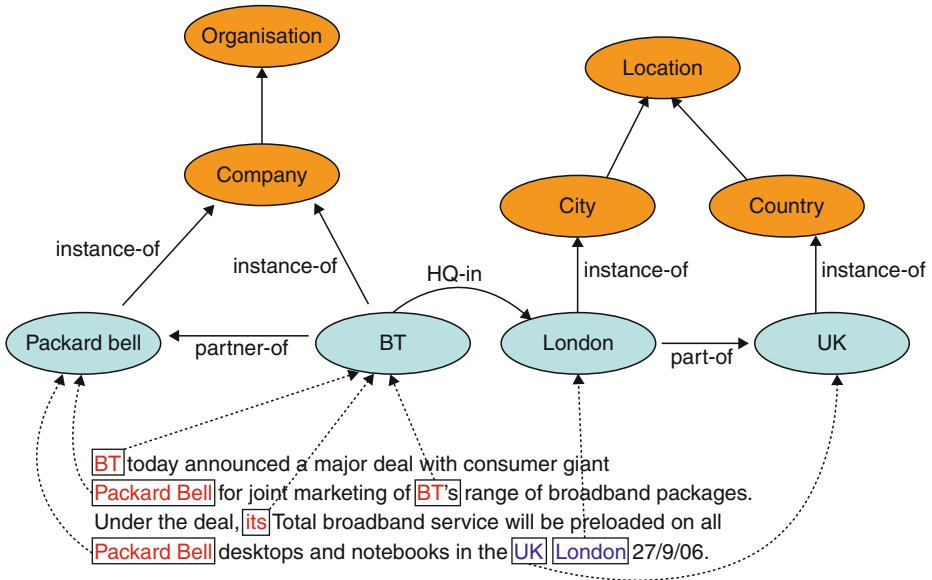
cope with synonymy. Moreover, the use of lexical concepts also enables hyponym-matching. A hyponym is a word of more specific meaning. Thus, referring to the example in ▶ Sect. 18.1.3.1, *CFO* is a hyponym of *director*. Hyponym-matching overcomes the problem that a search for *director* will not identify references to *CFO* which may be relevant.

Named entities are items such as proper nouns (denoting, for example, persons, organizations, and locations), numbers, and dates. One study found that named entities were a common query type, in particular people's names, while "general informational queries are less prevalent" [7]. Such named entities can be identified as instances of a predefined ontology. A typical ontology for such purposes would need to have information about people, geography, company structure, etc. One such ontology is PROTON [8], which was developed by Ontotext (<http://www.ontotext.com>) and used within the SEKT project (<http://www.sekt-project.com/>) as the basis for several semantic search and browse tools. In fact, PROTON also includes a world knowledge base. The word "knowledge base" is used to describe a set of instances and instantiated relations conforming to an ontology. Thus, the PROTON world knowledge base is a set of instances and instantiated relations, which are used to pre-populate the ontology. This initial knowledge base can then be extended through analysis of the textual corpus. Of course, this approach, while highly accurate, can lead to error. Therefore, information in the knowledge base is flagged to indicate whether it is predefined or whether it is learned from the document database. The PROTON ontology is itself extensible, any particular domain can develop its domain ontology as an extension to PROTON.

▶ Figure 18.1 illustrates how sentences can be analyzed and the named entities related to the classes of an ontology. Packard Bell and BT have been identified as instances of companies, while London and UK have been identified as instances of city and country, respectively. Once identified, these instances then form part of a knowledge base. Note that "its" has been identified as being equivalent to BT in this particular sentence. The identification of words such as pronouns with the words or phrases which they stand in for is known as anaphora resolution. Software to achieve this textual analysis is described in ▶ Sect. 18.2.5.1.

18.1.3.3 Storing Information for Easier Retrieval

Quite apart from the problem of finding information on the Web or corporate intranet, many people find it difficult to retrieve information they have stored on their personal computers. This subject has been extensively studied, for example, by William Jones [9, 10]. One reason for the difficulty is that people frequently do not have a consistently defined folder structure. In fact, even an entirely consistent structure can lead to ambiguity and questions such as "are the company financial results for 2008 in the folder *2008*, perhaps in a sub-folder *finance*, or in the folder *finance* in a sub-folder *2008*." Again, the problem is that the system is unable to understand semantics which are relatively obvious to a human, and which make it clear to the human that the paths *2008/finance* and *finance/2008* are likely to lead to related information. One proposed solution is the use of tags rather than folders. Reference [11] discusses the advantages and disadvantages of the two approaches.



■ Fig. 18.1

Relating the named entities in a sentence to an ontology

18.1.4 Sharing Knowledge Across the Organization

Sharing knowledge across large organizations is a notoriously difficult problem. Sometimes, the need is to make an employee aware of a document created by a colleague; at other times the need is to put the colleagues directly in touch. In any case, the colleagues may be completely unaware of each other and located geographically far apart. Of course, a useful document might have been created some time ago, by an employee who has moved on to other work or left the organization.

As already observed, consultancies such as Ernst and Young [12, 13], frequently take this subject most seriously. Typically they have a combination of part-time knowledge management enthusiasts in their operating units and full-time knowledge management specialists in a central unit. They use a platform, such as Lotus Notes, for document storage; a typical such document might be a customer proposal, which might be partially reused for other customers. In some cases, users may simply enter a document directly into the repository. In others, the document is vetted for quality by one of the knowledge management team. In both cases, the user will be required to describe the document using metadata compliant with a predefined taxonomy. Depending on the experience of the user and the particular document, this can take a significant amount of time and inhibit information being entered into the repository. A similar problem applies in reverse. To retrieve information, a user needs to understand the taxonomy, and of course the original metadata need to be accurate. Information may be missed, or the complexity of the system may again deter its use. What is needed is to analyze the documents as they are entered

into the system, so as to automatically create semantic metadata, which can be used for document retrieval. Automatic metadata creation also provides a consistency which may not occur when metadata are manually created.

Systems also exist for identifying people within the organization with a particular expertise. These may rely on employees inputting their information directly, with the result that the information is often not present or not up-to-date. Alternatively, they may use information collected by the human resource department, which has similar problems. What is really required is to understand a person's expertise by semantic analysis of the documents, e-mails, etc., which he or she creates and reads.

18.1.5 Helping with Processes

Current productivity tools offer basic support for processes, but little proactive help. Within Microsoft Outlook, for example, calendar and contact facilities provide tools for the user. However, all the intelligence needs to be supplied by the user. When the user types "phone John Smith" at a given time in his diary, there is no automatic link to the contact book entry for John Smith.

In addition, what information the system does have is routinely lost. Imagine the user receives an e-mail with attachments from John Smith as part of the customer X bid proposal process. He saves the attachments in a folder. Then the link between the attachments and John Smith, or customer X, is lost. If the user wants to find all information sent by John Smith or about customer X, then there is nothing associated with the saved files to help him. When he or she is working on the customer X proposal process, there are no metadata associated with those files to indicate their relevance to customer X.

Moreover, current systems have no idea of the context in which the user is working or the process currently being followed. For example, if the user is a patent lawyer with six different patent filings under consideration, the system has no idea which one is currently the focus of his attention. Nor does it know whether the user is creating a patent, reviewing a colleague's proposed filing, or searching for prior art. Yet such information would enable the system to proactively help the user. What is missing are metadata shared between applications and linked to the context of the user's work and the processes he or she performs.

18.1.6 Information Integration

18.1.6.1 The Challenges of Information Integration

McComb [14] suggests "that at least half the cost of integrating systems comes down to resolving semantic issues." Integration is a challenge in all organizations, but particularly where mergers and acquisitions have led to the need to rationalize different systems. Even without the stimulus of mergers and acquisitions, organizations often need to rationalize their information. For example, separate product lines may have different customer databases and this creates difficulties for cross-selling.

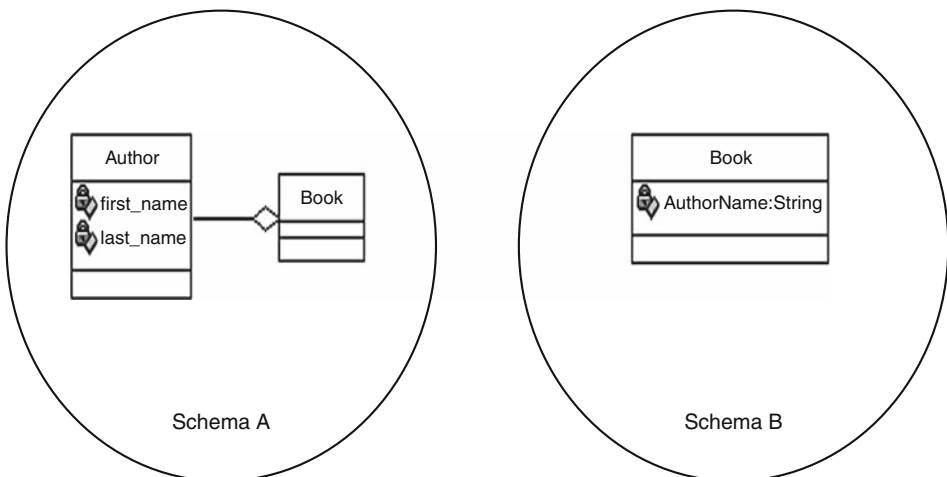
The problems of polysemy and synonymy, discussed earlier in a different context, arise again here. Different database schemas use the same terms with different meanings and different terms with the same meaning. Different database schemas may also use different structures and different values to represent the same information. This is illustrated in [▶ Figs. 18.2](#) and [▶ 18.3](#), which are adapted from [15]. These conflicts are very frequent, occurring as a natural consequence of data modeling – whether due to isolated development, changing needs, organizational or structural differences, or simply the different approach of two human data modelers.

As McComb points out, non-semantic issues such as language mismatch and platform boundaries, rarely cause surprise and can be planned for. It is the semantic mismatches which create the real problems in systems integration.

18.1.6.2 Approaches to Information Integration in the Enterprise

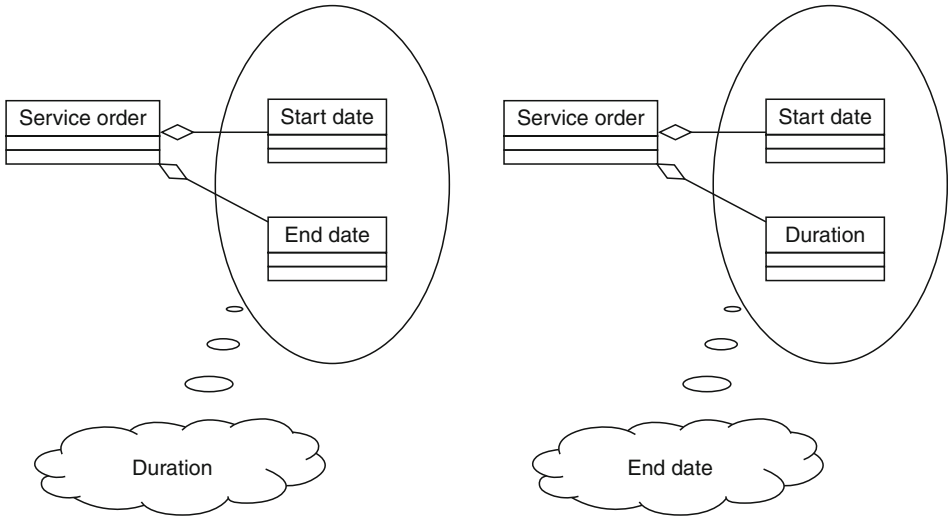
Information integration has been recognized as a significant problem in enterprises for some years, certainly well before the Semantic Web was conceived, and before the use of ontologies were a major subject of research in Computer Science. It is a problem of considerable economic importance. Based on a number of papers in the literature, Bernstein and Haas, claim that IT departments spend about 40% of their budget on information integration [16]. Their paper classifies information integration into a number of strands: data warehousing; virtual data integration; message mapping; object-to-relational mappers; document management; and portal management.

A *data warehouse* consolidates data from multiple database sources so as to allow querying to provide a comprehensive view of, for example, a customer. This consolidation



■ Fig. 18.2

Aggregation conflict – difference in structure



■ Fig. 18.3

Value representation conflict

is achieved through the use of Extract-Transform-Load (ETL) tools. The source databases are likely to have different schemas, and the warehouse database schema needs to permit mapping from each of these source schemas.

Virtual data integration avoids creating an actual warehouse of data yet also provides an integrated view. This is done by a query mediator, which translates the user's query into queries on the individual databases. Such an approach is referred to as Enterprise-information Integration (EII).

Message mapping uses message-oriented middleware to “integrate independently applications by moving messages between them.” Where a broker is used, this is called enterprise application integration (EAI) and where all applications use the same protocol this is called an enterprise service bus (ESB).

Data warehousing, virtual data integration, and enterprise application integration all involve mapping between database schema, which is the subject of this section.

Document management may be concerned with integration on a superficial level, for example, making documents available on a single *portal*. However, integration may also mean combining information from documents to create a new document or database.

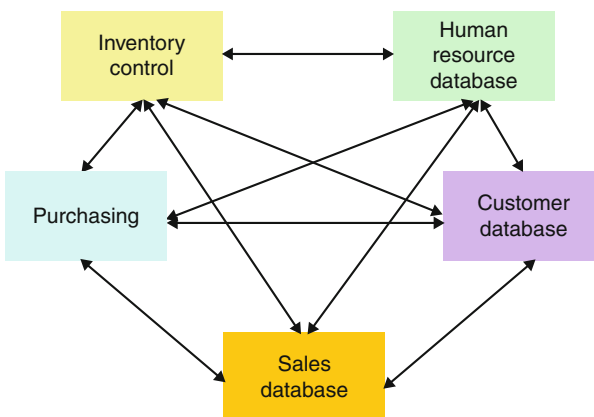
Bernstein and Haas, in their overview, also make the point that information integration was originally conceived as a predefined problem, that is, integrating a number of enterprise databases. More recently, the problem has widened, increasing to personal information management, creating a link with a theme of [Sect. 18.2.3](#).

For the personal views of a number of practitioners in the field of information integration, see [17]. Amongst the multiple authors, one (Pollock) argues strongly that EII will in the future make use of formal semantics. As Pollock sees it, the problem with database integration is that the structures contain no explicit formal semantics. Draper

stresses the importance of data modeling, and the need to model “the relationships and meaning of data separately from the aspect of when and where it is computed.” Rosenthal calls for, not just semantic integration but also “semantics management.” Within this, he includes guiding (e.g., enterprise managers) as to what concepts should be used, either to describe existing systems or for newly built systems. He sees this as a compromise between totally centralized and peer-to-peer systems. Bitton, arguing why EII will never totally replace data warehousing, draws attention to the performance implications of query processing in EII. Performance implications will remain an issue in the more sophisticated ontological approach proposed below. Finally, Sikka calls for a common semantic framework for information retrieval from structured and unstructured sources. This points again to the theme of [Sect. 18.2.3](#).

18.1.6.3 Using Ontologies for Information Integration

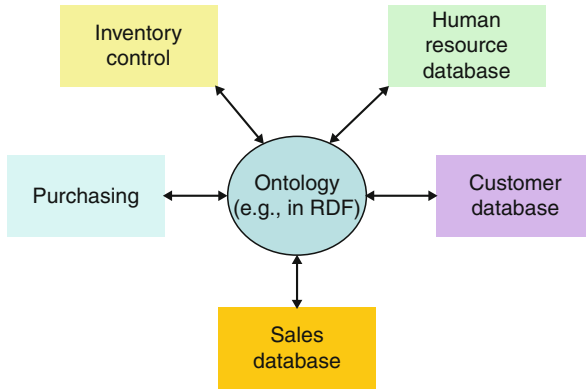
The value of ontologies in information integration stems from the ability to create an overarching ontology which can subsume multiple database schemas. The current state of the art in information integration is illustrated in [Fig. 18.4](#). To achieve integration at the semantic level, mappings are created between each database. These might be databases internal to one organization, for example, order processing and stock control databases; or the mappings might be across organizations, for example, between databases held by separate companies working together in a joint venture or supply chain. In any case, the problem is that the number of mappings increases quadratically with the number of databases.



N.B. This shows the situation within one organization. The problem multiplies when organizations wish to work together and link up their databases.

Fig. 18.4

Information integration today – many one-to-one mappings



■ Fig. 18.5

Illustrating the use of a central broker

► *Figure 18.5* illustrates the use of a central broker to reduce the number of mappings to that of the number of databases. Of course, the idea of a central hub is not new in systems integration. The innovation here is that the integration is at the semantic level, and is achieved through the use of a central overarching ontology based on open, lightweight standards. Note that the mappings are potentially both between schema and instances. To take two trivial examples, in the case of schema, a mapping is required which identifies “first name” and “forename” as the same; in the case of instances, the mapping must equate “Paul William Warren” with “Paul W Warren.”

Information about tools and techniques for creating semantic mappings is given later in ► [Sect. 18.2.4](#). An example of using this approach in the supply chain is described in [18]. The example shows how a number of Internet service providers can integrate their heterogeneous operational support systems with those of a telecoms operator, in this case BT. The approach reduces costs and time-to-market while, in particular the use of ontologies, enables a reuse of services.

18.1.6.4 Research Themes in Information Integration

There has been significant research activity into the use of ontologies for semantic integration. As long ago as 2004, there was a special issue of the ACM SIGMOD Record on Semantic Integration. In the introduction, the editors drew attention to three research activities, which remain challenges today [19]:

- Extending the scalability of schema techniques to large schemas.
- Designing the interaction with the user. It is generally accepted that a schema matching system will never be completely autonomous, and hence user interaction is required. The user interface has its own scalability problems. Moreover, schema matching may be part of a larger task, hence the schema-matching user interface needs to be embedded into some larger system.

- Mapping maintenance. Schemas change frequently, and therefore mappings need to be maintained.

The editors also noted the need for measures to establish similarity between schemas; similarity measures remain an active area of research.

In the same issue, Noy provides another view of the use of ontologies in semantic integration [20]. She divides research into semantic integration into three “dimensions”: mapping discovery, representations of the mappings, and reasoning with the mappings.

To facilitate mapping discovery, Noy argues for using common upper-level ontologies. She argues that “if two ontologies extend the same reference ontology in a consistent way, then finding correspondences between their concepts is easier.” Of course, this describes a situation where one is starting from scratch and extending an upper-level ontology to create domain ontologies. Where there are existing legacy ontologies, mapping will be much harder.

Turning to mapping representation, she identified three ways of doing this. One can construct an ontology of mappings, in which case the individual mappings become instances of concepts in the ontology. Alternatively, bridging axioms can be defined in first-order logic to represent transformations. Finally, views can be used to describe mappings from a global ontology to a local ontology, that is, the global ontology is used to provide access to local ontologies.

Another paper in the same issue emphasizes the need for customizability to create an “industrial strength” schema mapping tool [21]. The authors argue that customizability is needed to select and combine the techniques appropriate to the particular schema-matching problem; to control scalability, for example, by trading off response time and quality of the result; and to enable extensibility so that new techniques can be easily added. The authors also emphasize that schema matching is the first step in automating the creation of mappings between schemas. The second step is query discovery, in which queries are obtained to translate instances of the source schema into instances of the target schema. More recently, two of the authors of this paper, both at Microsoft, have gone on to describe their work in model management [22]. Model management is designed to support schema matching, merging, translation, comparison, and mapping composition. It is not a user-oriented tool, but rather a reusable component to be embedded in user tools. One aspect of the direction of this research is an increased emphasis on the runtime system to support the execution of mappings. The paper contains a review and comparison of, on the one hand, the approach focused on the mapping designer, and, on the other hand, their approach of focusing research on the model management. In fact, they see the two approaches as converging.

18.1.7 Integrating Structured and Unstructured Information

18.1.7.1 The Need to Analyze Text

Conventional corporate information systems are built on relational database technology. This is true whether the systems are for customer relationship management, product

information, employee information, competitor information, etc. Section 18.1.6 has just discussed the problems of integrating such database systems. A further problem lies in capturing unstructured information and semi-structured information. By “unstructured” information is meant information for which no schema exists, for example, information in text on the intranet, in memos on personal computers, in e-mails, slide presentations, etc. By semi-structured information is meant information for which some kind of schema exists but for which the schema is not defined as rigorously as is the case in relational databases. This includes information in applications such as spreadsheets where schemas may exist in the form of row and column headings.

The claim has been made that over 80% of the data in an organization is unstructured [23]. Whether this claim is true, or even practically verifiable, is not important. It is a common experience that a great deal of valuable information in an organization exists in this form. What is needed is to extract this information and transform it into structured form to enable merging with the structured data. The problem is that structured data have defined semantics in the form of schemas. These semantics may be local to the particular application, rather than being expressed using shareable ontologies, but they are semantics nevertheless. The application knows, for example, that the *price* field in a relational database contains the price in an agreed currency. In unstructured data it could be argued that the semantics are still there. A human can detect when a brochure describes a product price. However, the semantics are no longer defined in a machine-interpretable way. The price can be anywhere in the document and can be introduced by many different kinds of language. Interpreting these semantics is a task which until recently has been regarded as requiring human intelligence.

If structured information could be extracted from unstructured data, then there are many applications which would benefit. A complete picture could be built up, based on all the information available to the enterprise, of, for example, any particular customer, supplier, or competitor. Instead of searching separately through e-mails, memos, corporate intranet and databases, a sales advisor would have a complete picture of a customer, based on all those sources.

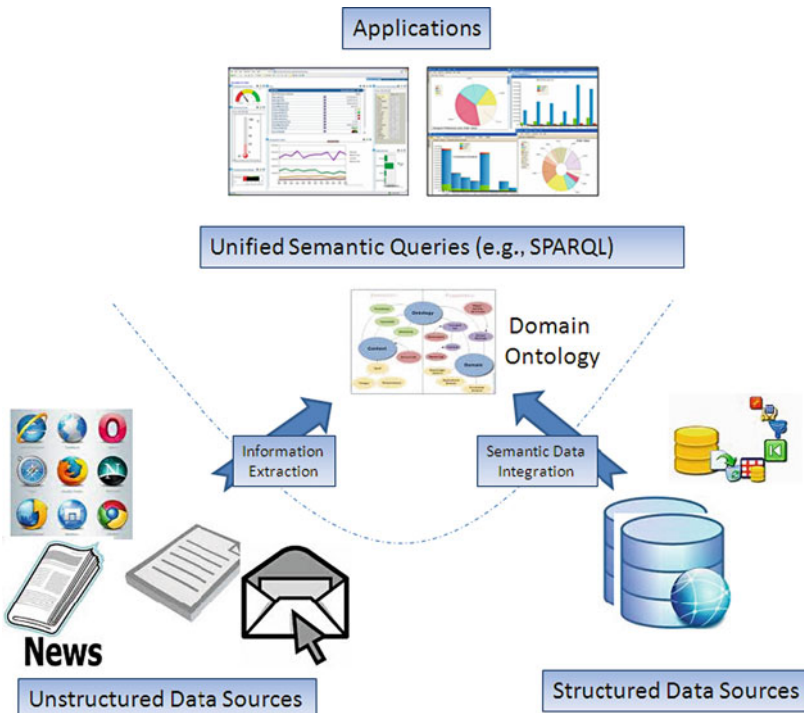
Added to the opportunity cost of not being able to use all the information potentially available to the organization, is risk associated with the regulatory environment. Organizations which do not disclose all relevant information to regulatory authorities may be seriously penalized. Yet the organization can only disclose information it knows it has. Information lost on corporate computers cannot be disclosed at the appropriate time – but will certainly be revealed if the organization is subject to a detailed forensic analysis of hard drives prior to a legal hearing. As an example, Forrester [24] describes a \$1.4 billion judgment against Morgan Stanley, arising from the latter’s inability to produce requested information.

The growing use of e-mail is one factor increasing the importance of unstructured information. AIIM (<http://aiim.org>), a nonprofit organization in the electronic content management industry, confirms that e-mail is a central means for business documentation [25]. Over 70% of the respondents to an AIIM survey reported exchanging

confidential or sensitive information via e-mail. AIIM found that e-mail is being used for critical processes such as contract negotiation, HR discussions, and invoice delivery. US public companies are also affected by the Sarbanes–Oxley Act, a US federal law enacted in 2002 which, among other things, sets enhanced reporting requirements for US public companies. Nearly one third of respondents reported that the Sarbanes–Oxley Act has affected the way their organization views e-mail.

All this points to a growing business need to understand the semantics of textual information, to extract such information from free text, convert into a structured form, and merge with preexisting structured information.

The overall goal is to combine structured and unstructured information and make the combined result available to a range of applications. This is illustrated in [Fig. 18.6](#) where information from a variety of unstructured sources is combined with information from databases to create information described in terms of an ontology. This can then be combined with domain-specific knowledge and business rules, and then operated on by semantic queries to input to client applications. Typical business rules would depend upon the application. For example, in sentiment analysis, where a company is interested in the perception of its products as expressed in blogs, etc., on the Web, then a rule might



■ Fig. 18.6

Combining structured and unstructured information

state that if customer perception for a particular product drops below a given level, then that product should be categorized as “at risk.” Combining information from structured and unstructured sources, a rule might say that if product sales have declined in the last month, compared with the month before, and customer perception has dropped below a particular level, then the product is in the “high-risk” category.

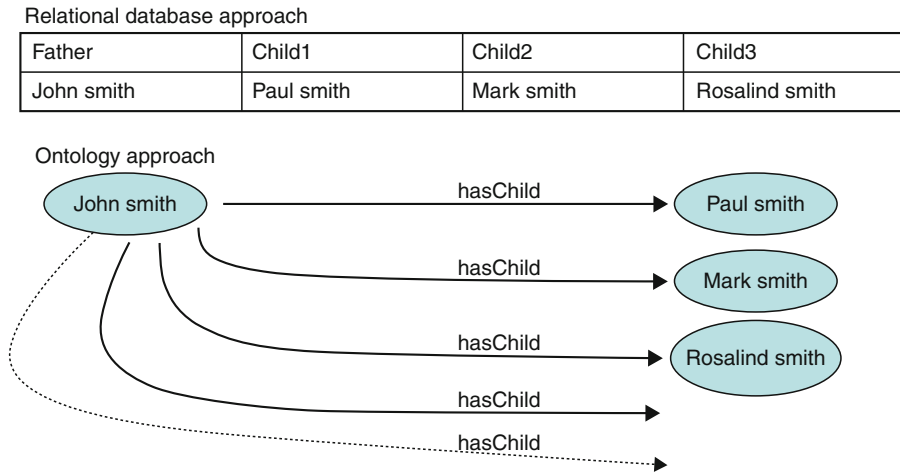
The essential challenge is to create some structure out of unstructured text. One way to do this is to create semantic metadata. HTML, the language which underlies the WWW and corporate intranets, is based on the use of metadata. However, the metadata in HTML are used to describe the format of data, for example, to indicate a heading or a bulleted list. The need here is to create semantic metadata, that is, metadata which provide information about the data.

Such metadata can exist at two levels. They can provide information about a document or a page, for example, its author, creation, or last amendment date, or topic; or they can provide information about entities in the document, for example, the fact that a string represents a company or a person or a product code. The metadata themselves should describe the document or entities within the document in terms of an ontology. At the document level, there might be a property in the ontology, for example, has Author, to describe authorship. Within the document classes such as Person, Company, or Country would be used to identify specific entities.

18.1.7.2 Combining the Statistical and Linguistic Approaches

The metadata could be created by the authors of the document. In general, this will not happen. The authors of Word documents or e-mails will not pause to create metadata. The need is to generate metadata automatically, or at least semiautomatically. There are two broad categories of technology which can be used for this: statistical or machine learning techniques; and information extraction techniques based on natural language processing. The former generally operate at the level of documents, by treating each document as a “bag of words.” They are, therefore, generally used to create metadata to describe documents. The latter are used to analyze the syntax of a text to create metadata for entities within the text, for example, to identify entities as Persons, Companies, Countries, etc. Nevertheless, this division should not be seen too starkly. For example, one of the goals of the SEKT project (<http://www.sekt-project.com>), a European collaborative research project in this area which ran from 2004 to 2006, was to identify the synergies which arise when these two different technologies are used closely together. An overview of semantic knowledge management, including these two approaches to creating metadata, is given in [73].

The metadata can create a link between the textual information in the documents and concepts in the ontology. Metadata can also be used to create a link between the information in the document and instances of the concepts. These instances are stored in a knowledge base. Thus, the ontology bears the same relationship to the knowledge base as a relational database schema bears to the information in the database. In some cases, the



In a database there are a predefined number of fields, e.g., here there are three “child” fields. An ontology is more flexible, e.g., one can have an unlimited number of instantiations of the “hasChild” property.

■ Fig. 18.7

Ontologies offer greater flexibility than database schema

ontology and the knowledge base will be stored together, in other cases separately. This is essentially an implementation decision.

Ontologies are particularly useful for representing knowledge from unstructured text because of their flexibility and ability to evolve. Once created, ontologies can be far more easily extended than is the case for relational database schema. ▶ *Figure 18.7* provides a simple illustration of how the ontological approach overcomes the limitation in databases of having a predefined number of fields. Here, the occurrence of new children simply requires new instantiations of the “hasChild” relation. This contrasts with a database design where one would need to decide at the beginning the maximum number of children a person might have. Moreover, it is not even necessary to decide initially what relations are needed. The ontology designer might realize at some stage that the “hasBrother” relation is useful in some cases. This can be added to the ontology far more easily than adding a new field to a database. This is not to say that the ontology-based approach will replace the use of relational databases. With increased flexibility comes increased computational expense. The ideal is to combine the two approaches.

Where the system identifies a text string as an instance of a concept in the ontology but which is not represented in the knowledge base, then that instance can be added to the knowledge base. For example, the text string “ABC Holdings” may be identified as a company, but one not represented in the knowledge base. The system can then add “ABC Holdings” to the knowledge base. ▶ *Section 18.1.3* has already discussed how entities in text can be associated with entities in the knowledge base; this was illustrated in ▶ *Fig. 18.1*.

Research is also in progress to use natural language processing techniques to learn concepts from text, and thereby extend the ontology. However, this is a significantly harder problem. For an example of the state of the art, see [74].

18.1.8 Sharing Knowledge Between Organizations

There are a number of motivations for an organization wanting to share knowledge with other organizations. One of the most obvious is to cooperate in a supply chain, where the information shared is contractual. Another is to undertake collaborative research, or simply to share research results. A discussion of knowledge sharing in the supply chain is properly the domain of eBusiness, which is discussed in the next chapter of this volume; while knowledge sharing for research is the domain of eScience, discussed in the previous chapter. However, there are situations where organizations need to collaborate together to achieve common goals, and where the activity might properly be regarded as knowledge management. One such is within the domain of medicine, where general practitioners and clinicians need to share information about patients, for example, describe their symptoms. Of course, the boundary between eBusiness, eScience, and knowledge management is somewhat fuzzy. In the medical example, the same vocabulary might be used in a clinical environment (knowledge management), to share information with an insurance company (eBusiness), or for research into illness (eScience).

In any case, the problems are similar. There is a need for a shared vocabulary, for example, for use within an industry sector or within a specialism. Usually these vocabularies, created and maintained by a standards body, are defined in a natural language, frequently English. Such informal definitions give rise to redundancies and even inconsistencies. They also give rise to misunderstandings when different parties interpret the natural language differently. What is required is a more formal approach based on knowledge representation techniques, for example, ontologies. The use of the informal approach is partly historical, some of these vocabularies have a long history going back before the use of ontologies was proposed. Even today, many of the people developing such vocabularies will not be skilled in knowledge representation and will use natural language. As a consequence, it is frequently necessary for ontologists to come along after the event and create a more structured approach out of what exists informally. This is true in eBusiness where Electronic Data Interchange standards such as ANSI ASC X12 (<http://www.x12.org/>) and the United Nations's EDIFACT (e.g., <http://www.unece.org/trade/untdid/welcome.htm>) have been in existence for some decades. An attempt to use an ontology to describe at least the syntax of X12, prior to "ontologizing" the semantics, is described in [26]. [Section 18.2.6.1](#) described an example more properly from knowledge management, that of the use of ontologies in medical informatics.

An alternative approach to shared vocabularies is to use, for example, RDF, to create self-describing data and to make that data available to other organizations. If that data is made openly available on the Web, then this creates a Web of linked open data. This is

exactly what the *linking open data* initiative is in the process of achieving; this is described briefly in [▶ Sect. 18.2.6.2](#) and in more detail in [▶ Semantic Annotation and Retrieval: Web of Data](#).

18.1.9 Another Look at Ontologies

The constant theme running through this chapter has been the use of ontologies. An early, but still relevant, overview and categorization of the ways ontologies can be used for knowledge sharing is given in [27]. Here, the use of ontologies is categorized in a number of ways. Ontologies can be used in conjunction with conventional (i.e., nonintelligent) software or alternatively in conjunction with software employing AI techniques. The reference lists a number of principles which remain true: knowledge engineering needs to be minimized, as it represents an overhead; KM support needs to be integrated into everyday work procedures; and KM applications need to process information in an integrated manner. It describes a range of applications which remain important: knowledge portals for communities of practice; lessons learned archives; expert finders and skill management systems; knowledge visualization; search, retrieval, and personalization; and information gathering and integration.

Another high-level view of ontologies, and specifically their use in achieving data connectivity, is given by Uschold and Gruninger [28]. They note that connectivity is required at three layers: physical, syntactic, and semantic. Great strides have been made in achieving connectivity at the first two layers. The challenge is now the third, and ontologies have a key role here. Semantic heterogeneity is a fact of life to be overcome – “there will always be sufficiently large groups for which global agreements are infeasible.” They present a spectrum of kinds of ontologies, defined by degree of formality. At the informal end, there are sets of terms, with little specification of the meaning, and also ad hoc hierarchies, such as in Yahoo!. At the formal end, there are, for example, description logics. At the informal end, some of these might not properly be called ontologies, for example, by members of the knowledge representation community. The point is that they are used in similar ways as some formal ontologies. Uschold and Gruninger compare ontologies with database schema; making the point that the mixing of types (concepts) with instances is a feature of ontologies which does not occur in database schema. In their view this is largely because of the much greater scale and performance requirements for database systems. Note that this is a computational feature; computationally database schema and database instances are treated quite separately. This is less the case in the ontological approach; indeed it can in some cases be a matter of design style whether an entity is represented as a concept or an instance. However, when one turns to implementation, the converse can be true. A database schema is embedded in the database; an ontology can exist in a separate physical implementation.

The authors of this chapter have prepared their own summary of the chief differences between the relational database and ontological knowledge base approach. This is summarized in [▶ Fig. 18.8](#).

	Relational database	Ontological knowledge base
Information model	Schema <ul style="list-style-type: none"> • Hard to evolve • Implemented with instances in database • Computationally separate from instances 	Ontology <ul style="list-style-type: none"> • Flexible • Can be implemented separately from instances • Computationally concepts and instances treated similarly
Information which can be retrieved	What you put in is what you get out	Information entered into knowledge base plus inferences from that information

■ Fig. 18.8

Comparison of relational databases and ontological knowledge bases

Uschold and Gruninger identify four ways in which ontologies help achieve a common understanding. Three are relevant to the theme of this chapter:

- Neutral authoring. Here an ontology exists for authoring purposes, and the results are then translated into a variety of target ontologies. Enterprise modeling is an example of this.
- Common access to information. Here, the ontology is used as a neutral interchange format, as discussed above. The objective is to avoid the need for $O(N^2)$ translators.
- Query-based search, that is, a sophisticated indexing mechanism with the added benefit of permitting answers to be retrieved from multiple repositories.

Uschold and Gruninger describe the first of these as using neutral ontologies, without describing formally what the adjective “neutral” means here. They go on to add that, in the case of neutral authoring, the ontology can contain only those features present in all of the target systems and that, in the case of providing common access to information, the neutral ontology must cover all of the concepts in each of the target systems. This, in a sense, provides a definition of what “neutral ontology” means in each of these two cases. In the latter case what Uschold and Gruninger call a neutral ontology is what others refer to as an overarching ontology.

They also identify the use of ontologies for specification in software engineering, which is beyond the scope of this chapter.

18.2 Example Applications

Building on the discussions in ▶ Sect. 18.1, this section describes example applications of semantic technology addressing each of the challenges described in the previous section. ◀ Sections 18.2.1–18.2.6 describe responses to each of these challenges: searching and finding information; sharing information within organizations; helping users to navigate processes, including by taking account of the user’s context; integration of structured data; extraction of structured information from unstructured data; and sharing information across organizations.

18.2.1 Semantic Search, Browse, and Information Storage

18.2.1.1 Squirrel: An Example of Semantic Search and Browse

Squirrel [29] provides combined keyword-based and semantic searching. The intention is to provide a balance between the speed and ease of use of simple free text search and the power of semantic search. In addition, the ontological approach provides the user with a rich browsing experience. For its full-text indexing, Squirrel uses software from the open-source Lucene suite, see <http://lucene.apache.org/>. PROTON is used as the ontology and knowledge base, while KIM [30] is used for massive semantic annotation.

The KAON2 [31] ontology management and inference engine provides an API for the management of OWL-DL and an inference engine for answering conjunctive queries expressed using the SPARQL syntax. KAON2 also supports the Description Logic-safe subset of the Semantic Web Rule Language (SWRL). This allows knowledge to be presented against concepts that goes beyond that provided by the structure of the ontology. For example, one of the attributes displayed in the document presentation is “Organization.” This is not an attribute of a document in the PROTON ontology; however, affiliation is an attribute of the Author concept and has the range “Organization.” As a result, a rule was introduced into the ontology to infer that the organization responsible for a document is the affiliation of its lead author.

Users are permitted to enter terms into a text box to commence their search. This initially simple approach was chosen since users are likely to be comfortable with it due to experience with traditional search engines. Squirrel then calls the Lucene index and KAON2 to identify relevant textual resources or ontological entities, respectively. In addition to instance data, the labels of ontological classes are also indexed. This allows users to discover classes and then discover the corresponding instances and the documents associated with them without knowing the names of any instances, for example, a search for “Airline Industry” would match the “Airline” class in PROTON. Selecting this would then allow the users to browse to instances of the class where they can then navigate to the documents where those instances are mentioned.

➤ *Figure 18.9* shows the meta-result page. This is intended to allow users to quickly focus their search as required and to disambiguate their query if appropriate. The page presents the different types of results that have been found and how many of each type for the query “home health care.”

➤ *Figure 18.10* shows a document view. The user has selected a document from the result set, and is shown a view of the document itself. This shows the metadata and text associated with the document and also a link to the source page if appropriate – as is the case with Web pages. Semantically annotated text (e.g., recognized entities) is highlighted. “Mousing over” recognized entities provides the user with further information about the entity extracted from the ontology. Clicking on the entity itself takes the user to the entity view.

➤ *Figure 18.11* shows an entity view for “Sun Microsystems.” It includes a summary generated by OntoSum [30]. OntoSum is a Natural Language Generation (NLG) tool which takes structured data in a knowledge base (ontology and associated instances) as

Matches for your query:

[Journal Articles](#): 76

[Conference Papers](#): 46

[Periodicals](#): 257

[Web Pages](#): 16

[Library Topics](#) (60) including: [Home health care](#)(4), [Health care \(Technical\)](#)(135), [Rural health care](#)(1), [Mental health care](#)(4), [Long term health care](#)(2)

[Organisations](#) (597) including: [National HealthCare Corporation](#)(PublicCompany), [National Home Health Care Corp.](#)(PublicCompany), [OhioHealth](#)(Company), [St. Luke's Episcopal Hospital](#)(Company), [Sunquest Information Systems, Inc.](#)(PublicCompany)

[Knowledge Base](#) (673) including: [National HealthCare Corporation](#)(PublicCompany), [Home Health Care Services](#)(IndustrySector), [Home Health Care Services](#)(IndustrySector), [National Home Health Care Corp.](#)(PublicCompany), [OhioHealth](#)(Company)

Fig. 18.9

Meta-results page

Document: Helping employees stay healthy

Abstract: About 240 big companies (which together provide health insurance for more than 50 million Americans) have banded together to create a nonprofit organization called the **National Business Group on Health**. It advises large employers on health-care and benefits issues and has started bestowing a new honor - the **Best Employers for Healthy Lifestyles** award - on companies that are

Author: **Fish** National Business Group, Inc. is a Company located in United States (North America). Its webpage is <http://www.nbg.com>. National Business Group (NBG) knows it often takes networking to get a job. The systems

Topics: **Em** integrator specializes in LANs, WANs, remote access, and network security. NBG also resells networking and

Date Published: **Aug** connectivity equipment and offers network analysis, consulting, design, installation, technical support, and training services. The company purchases from such suppliers as Cisco Systems, and Microsoft, and Nortel

Full Text: **EVE** are in the U.S. and of how worried
em e seen the fallout close to home, in the
form nd copayments. But corporate **America** is
final s (which together provide health insurance
fin r to create a nonprofit organization called
for employers on health-care and benefits
the Employers for Healthy Lifestyles award-on
iss thier.
con

Some of the winning tactics include installing walking routes and hiking paths around workplaces and stocking cafeterias and vending machines with more fruit and other foods containing less fat and salt. NBGH hopes that highlighting such best practices will encourage other employers to copy them. Additional praised tactics:

Fig. 18.10

Document view

input and produces natural language text, tailored to the presentational context and the target reader. NLG can be used to provide automated documentation of ontologies and knowledge bases and to present structured information in a user-friendly way.

The summary displays information related not only to the entity itself but also information about related entities such as people who hold job roles with the company. This avoids users having to browse around the various entities in the ontology that hold relevant information about the entity in question.

Users can choose to view results as a consolidated summary (digest) of the most relevant parts of documents rather than a discrete list of results. The view allows users to read or scan the material without having to navigate to multiple results. Figure 18.12 shows a screenshot of a summary for a query for “Hurricane Katrina.” For each

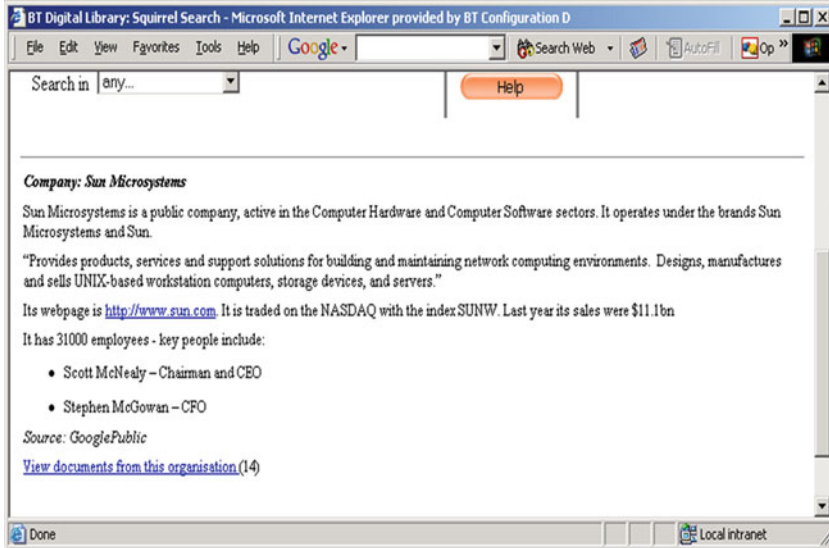


Fig. 18.11

Entity view

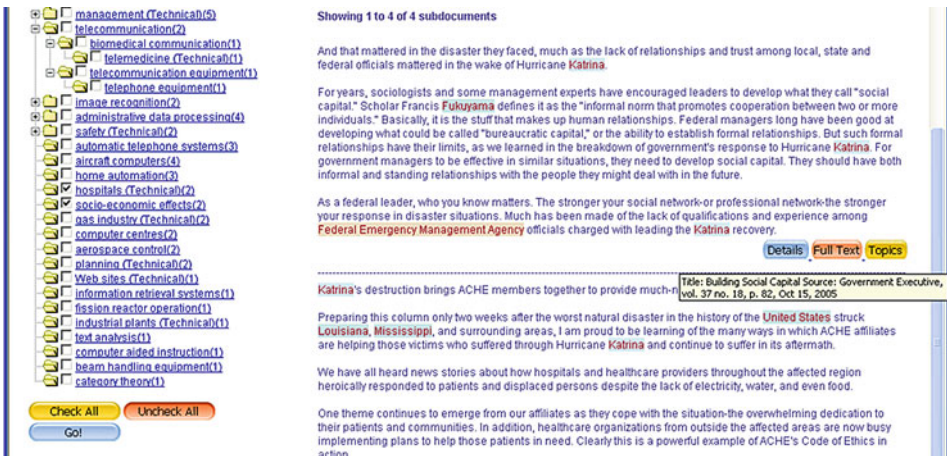


Fig. 18.12

Consolidated results

subdocument in the summary, the user is able to view the title and source of the parent document, the topics into which the subdocument text has been classified or navigate to the full text of the document. The example of Squirrel shows that not only does semantic search offer the potential to improve search results, but also to improve the presentation of those results.

To gain an idea of how users perceive the advantages of semantic search over simply text-based search, Squirrel has been subjected to a three-stage user-centered evaluation process with users of a large Digital Library. Twenty subjects were used, and the perceived information quality (PIQ) of search results obtained. Using a seven-point scale the average (PIQ) using the existing library system was 3.99 compared with an average of 4.47 using Squirrel – a 12% increase. The evaluation also showed that users rate the application positively and believe that it has attractive properties. Further details can be found in [32].

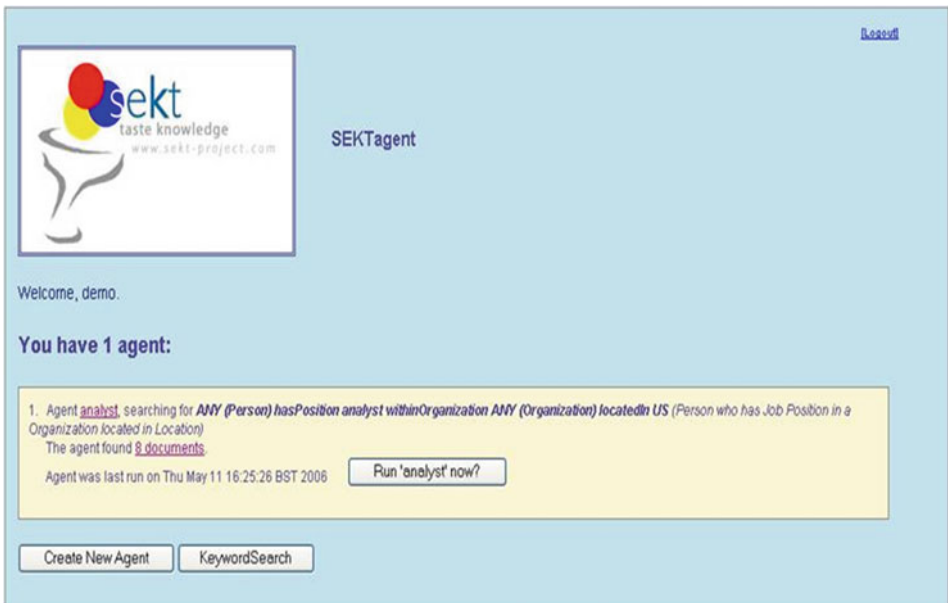
18.2.1.2 SEKTagent: A Different View on Semantic Search

Another approach to enabling semantic queries is exemplified by SEKTagent [29].

► *Figure 18.13* illustrates the basic approach by showing the following semantic query:

“ANY (Person) hasPosition analyst withinOrganization ANY (Organization) locatedIn US”

The query is looking for someone who is an analyst working in any US organization. This is quite different from a text query. Everything is stated at a conceptual level. The most concrete entity in the query is “US.” However, even this is not treated as a text string. The query may find a document referring to an analyst working in some city or state of USA, but not containing any reference itself to USA. The system makes use of the geographical knowledge in the knowledge base to determine that this is a relevant document.



The screenshot shows the SEKTagent web interface. At the top left is the logo for 'sekt taste knowledge' with the website 'www.sekt-project.com'. The title 'SEKTagent' is displayed in the top right. Below the logo, it says 'Welcome, demo.' and 'You have 1 agent:'. A yellow box contains the following information: '1. Agent analyst, searching for *ANY (Person) hasPosition analyst withinOrganization ANY (Organization) locatedIn US (Person who has Job Position in a Organization located in Location)*. The agent found 8 documents. Agent was last run on Thu May 11 16:25:26 BST 2006'. There is a button labeled 'Run 'analyst' now?'. At the bottom, there are two buttons: 'Create New Agent' and 'KeywordSearch'.

■ **Fig. 18.13**

A semantic query in SEKTagent

for such tasks, said Gartner analyst Kimberly Harris-Ferrante. However, she said
nce company's lack of experience with large outsourcing deals.

■ **Fig. 18.14**

Extract from one of the results of a semantic query – showing entities in the knowledge base highlighted

► *Figure 18.14* shows an extract from one of the retrieved documents. Entities in the knowledge base are highlighted. In this case, there are three such entities: Gartner; analyst; Kimberley Harris-Ferrante. The first of these is a company, the second a position in an organization; and the third is a person. In fact, Kimberley Harris-Ferrante is the analyst, working in a US organization, who satisfies this query.

Moving the mouse over any of these entities displays more information about them. In the case of Gartner, for example, it provides the key facts about the company. Rather than just displaying raw information, natural language generation technology is applied to the relevant information in the knowledge base to create text, which can be easily read.

The example illustrates another important feature which differentiates the ontology-based approach from that of relational databases. In a database, the only information which can be retrieved is that which is explicitly input into the database. An ontology-based system can make use of a reasoner to perform inferencing over the ontology and knowledge base. In the example, the request was for someone performing a specific role in an organization in the USA. The information in the knowledge base could well be that the organization is located in some part of the USA, e.g., a city or state. However, the knowledge base associated with PROTON also has geographical information including states and major cities in the USA. Armed with this information, it is able to make the necessary inferences.

It should be noted that to identify any named geographical region (such as a county, state, region, district, town, village, etc.) with a particular country is in the general case a hard problem. However, a subset of the problem can be solved based on the knowledge available in the ontology. For example, the PROTON ontology contains, for all major cities, a link to the country in which they are located. It is relatively easy therefore to identify a major city in the query and link it with the appropriate country. In other applications, more domain-specific information may be required; frequently, it may be possible to draw on information already in structured or semi-structured form and thereby reduce the need for manual intervention.

Additional examples of semantic search are given in [18].

18.2.1.3 Semantic Filing: TagFS and SemFS

► **Section 18.1.3** discussed the difficulty which many people have in finding information which they themselves have stored, often on their own computers. One reason for this is that there is often more than one location where a file can logically be stored; yet users are

in general restricted to storing information in a single location. A partial solution to this is the use of tags. However, this loses the advantage of being able to travel through the tree structure of a hierarchical set of folders.

TagFS [33] merges the two approaches to obtain the advantages of both by using the tags to create a folder structure, which is dynamic rather than fixed. In TagFS, the organization of the resource is divorced from its location. The file is simply tagged. To take the example from the reference, in a conventional filing system, a user saving music files would first establish a directory structure, for example, *year/artist/album*. This would be quite distinct from a structure *artist/album/year*. In TagFS these three attributes, and any other which are appropriate are merely used to motivate tags. To find a file, it does not matter in which order you traverse the “directory”; the “directory path correspondingly denotes a conjunctive tag query which results in a set of files that fulfill all tag predicates.”

Apart from overcoming the need to specify folders in a specific order, tagging has the advantage that the user does not need to reach the end of a folder path before finding the required file. In addition, new tags can be added to describe a file in a way which new folders cannot.

TagFS is implemented using the SemFS architecture. SemFS provides mapping from traditional file system interfaces to annotation of information objects using RDF. Rather than interpreting directory structures as static storage hierarchies, as in a conventional file system, they represent dynamic views on information objects. In fact, TagFS makes relatively simple use of SemFS, in that the latter offers an arbitrary number of different views, while TagFS simply employs one called “hasTag.” The use of RDF enables integration with other semantic desktop applications, as described in [▶ Sect. 18.2.3](#).

18.2.1.4 Commercial Activities

There are a range of companies in this area, with new companies joining some established ones. In the domain of semantic search, there are companies such as Hakia, PowerSet (now acquired by Microsoft), Siderean, and Ontotext. In the information and process integration space there are, for example, Metatomix and Ontoprise. Turning to social networking and knowledge management generally, a company which has attracted recent interest is Radar Networks. In 2007, they announced their Twine semantic social networks offering. Twine mined fora, wikis, databases, and online newsgroups to identify relationships which were then expressed in RDF. Recently Radar Networks were acquired by Evri, and currently Twine is not supported. Evri themselves offer a “discovery engine” which identifies the currently most popular stories and trends.

Larger, more established vendors are also active, including Oracle with RDF support in Oracle 10g and ThomsonReuters making all their information available with semantic markup via their OpenCalais (<http://www.opencalais.com/>) service, which parses text for names, locations, organizations, and other entities.

In the search sector, PowerSet, mentioned above, was acquired by Microsoft for \$100m. Microsoft is believed to have incorporated aspects of PowerSet’s semantic

technology into its Bing Search engine. Yahoo! and Google have been more explicit in their use of semantic technology: Yahoo!'s Search Monkey platform allows developers to exploit semantic data (in RDFa or microformats). The idea is to make Yahoo! Search results more useful and visually appealing, and thereby drive more relevant traffic to their sites. In addition to the possibility for developers to create their own enhanced results, Yahoo! already provides a standard enhanced result for those sites providing structured data. Google followed suit with a similar initiative, known as Rich Snippets.

18.2.2 Semantic Information Sharing

▶ [Section 18.1.4](#) identified the importance, particularly acute in large organizations of being able to share information among colleagues. This applies both to knowledge explicitly written down and to tacit knowledge. In the former case, the need is to identify a document; in the latter case a person.

18.2.2.1 Effective Document Sharing with Semantic Technologies

Using Taxonomies for Knowledge Sharing

One way to share documents is simply to use the corporate intranet as a repository and provide employees with an intranet search engine. As already noted, search technology is not always fully effective. Even with the kind of advanced search technology discussed in ▶ [Sect. 18.2.1](#), a relevant document may be missed. One solution to make it easier to find and reuse documents is to require the author of the document to associate metadata with it when committing the document to a repository. Typically, the metadata relate to an agreed taxonomy.

As already discussed, the problem with this approach is that it can be time consuming for an author to save a document to the repository. The time taken will depend on how familiar he or she is with the system and the taxonomy, and also on the nature of the document. Frequently the time required is an inhibitor and the document will not be saved. A means of overcoming this is described in [\[34\]](#). Machine-learning techniques are used to automatically suggest metadata to the user, who can accept the suggestion, or make amendments or additions. The metadata can then be used by other users to search and browse the repository. Since this requires knowledge of the taxonomy, the system also offers a natural language search which requires no prior knowledge on the part of the user of how the information is classified.

A commercial example of a taxonomic system which offers support to the user is provided by Teragram (<http://www.teragram.com/>). The system employs linguistic technology. For example, an administrator is able to create rules to define which documents fall into each category of a taxonomy tree. Alternatively, the administrator can assign initial documents to each category and the system can then automatically make further assignments.

Using Ontologies

Taxonomies are limited in their descriptive power to describing hierarchical relationships. Ontologies are much richer in what they can describe. They offer an obvious basis for describing, and hence sharing information.

However, because of this increased richness, ontologies are in general more complex, and hence their creation and maintenance may be more time consuming. This depends, of course, on the tools available and the application domain. Similarly, from the user's viewpoint, the ontological approach will often be more time consuming than the taxonomic one. Once again, the kind of semantic annotation techniques described in [Semantic Annotations and Retrieval: Manual, Semiautomatic, and Automatic Generation](#) of this handbook can be used to automate, or at least partially automate, this process. The user wishing to retrieve information is then able to use the semantic search and browse techniques described in [Sect. 18.2.1](#). Reference [35] describes an implementation of this approach in a digital library. Here annotation is at two levels. Firstly, sets of topics are used to describe documents. Topics can have sub- and super-topics, to create a lattice structure. As a design decision, for reasons of computational tractability, topics are implemented as instances, not concepts. As a starting point, schemas used by proprietary information providers (e.g., Inspec: <http://www.theiet.org/publishing/inspec/>) provided the topics. Machine learning was used to refine these topics and to automatically associate documents with topics. Secondly, using natural language techniques, named entities within documents are identified and associated with concepts. These concepts are drawn from, for example, geography and business and include country, city, company, CEO, etc. The association of instances to concepts is illustrated by color coding, using the KIM system described in [36].

The creation and management of ontologies is required for many applications of semantic technology and is a significant research topic in itself. An overview of available methodologies is given in [37], which also describes a methodology, DILIGENT, for creating and maintaining distributed ontologies. In common with other such methodologies, the approach employs ordinary users, domain experts, and experts in ontology design. The approach is distributed in that different users may have slightly different versions of the ontology. Users refine a shared ontology on the basis of their experience, and these refinements are then fed back, as appropriate, to the shared ontology.

Tagging and Folksonomies

In parallel to the use of taxonomies in enterprises, and research into the use of ontologies, the hobbyist and consumer world has adopted the use of informal tagging to describe all kinds of information and media objects. Such tags are said to constitute “folksonomies.” Like wikis, folksonomies are part of the phenomenon of Web2.0, in which consumers of information are also producers. Such folksonomies are commonly represented by “tag clouds,” in which character size, font, or color are used to represent how much the tag has been used. Flickr (<http://www.flickr.com>) is an example of a website for sharing photos which uses this approach. Delicious (<http://delicious.com>) is another example where tags are associated with bookmarked pages. The website displays not just the most popular bookmarks, but also the most popular tags.

Some organizations now use similar techniques to encourage knowledge sharing and a McKinsey survey of the use of Web2.0 in companies has shown that many executives do believe that these techniques provide real business benefit [38]. Folksonomies have the advantage over taxonomies and ontologies in that they are easy to use. They do not have the development and maintenance costs associated with the use of the taxonomies and ontologies, that is, the cost of creating the taxonomy or ontology and then creating and updating the associated metadata.

McKinsey considered a range of Web2.0 technologies, including videosharing, blogging, RSS, wikis, and tagging. They looked at three broad areas of application: within organizations, which has been the theme of most of this chapter; in their dealings with suppliers and partners, which is outside the scope of this chapter; and in their relations with customers, which is similarly outside of the chapter's scope. They asked respondents to quantify the business benefit of using Web2.0 tools for each of these three areas. They found the median increase in speed of access to knowledge to be 30% and the median increase in speed of access to internal experts to be 35%. Other benefits were reduced communication and travel costs and reduced time to market. Similar responses occurred when respondents were asked about the effect of Web2.0 on collaboration with partners and suppliers. Not surprisingly, high technology and telecommunication companies reported the highest benefits with "business, legal, and professional services" also reporting a high level of benefits and manufacturing and financial further behind. Even so, in all industry sectors over 50% of respondents reported at least one measurable benefit from using Web2.0 technologies.

However, folksonomies lack descriptive power. In general, they possess no structure, usually not even the hierarchical structure present in a taxonomy. Moreover, the problems of synonymy and polysemy occur here; the same tag may be used with different meanings, or different tags may be used with the same meaning. Compared with ontologies, folksonomies are even more limited. They do not permit automated reasoning, nor the kind of search and browsing techniques described earlier. In general, the user is free either to use a preexisting tag or to use a new tag. The former has the practical value of encouraging convergence on a reasonable number of tags. However, it may lead to the emergence of dominant tags, representing particular views, and discourage the creation of new tags which may better represent a concept.

Nevertheless, after the success of tagging in the hobbyist world, it was natural to investigate the same approach in the enterprise. IBM's Dogear [39] is a bookmarking system in which bookmarks can be tagged. Once created, tags can not only be used for searching and browsing, but also to support social networks. The IBM designers of Dogear specifically chose to use real names, rather than pseudonyms. It is therefore possible for other users to see who has bookmarked a particular document. Knowledge of the particular bookmarks browsed by a user provides information about the user's expertise, or at least interests. This, in turn, enables the creation of communities of interest and potentially the identification of experts.

Another approach is to combine the ease-of-use of the folksonomic approach with the greater power of taxonomies. Reference [40] describes a proof-of-concept system which

suggests tags to the user by automatically selecting terms from a taxonomy. The user is, however, free to use other tags, and these are fed back to suggest new terms for the taxonomy. The authors call this a “taxonomy-directed folksonomy.” When users type a tag, they are prompted by a thesaurus which suggests terms which match the term they have entered. In principle, users could be given a choice of thesauri for tagging.

Heymann and Garcia-Molina [41] have developed an algorithm which converts a tag cloud into an hierarchical taxonomy. The starting point is to create a *tag vector* for each tag, of dimensionality equal to the number of objects, and such that the component in each dimension is the number times the tag has been applied to a particular object. From this, the cosine similarity between tag vectors is used to calculate the similarity between tags. These similarities are used by the algorithm to create a taxonomy.

Other work has combined user tagging and an ontology-based approach to the classification of information [42]. The goal of this work was to share information, in the form of bookmarked Web pages, and also to enable users to gain an awareness of others’ interests and expertise. Web pages are automatically classified, on the basis of their content, according to a preexisting library ontology. They are also tagged informally by users. A persistent problem with tagging is that different users will use different tags for the same concept, and the same tag for different concepts. In this work, equivalences are learned between different users’ tags on the basis of the content tagged. Moreover, the system recognizes relationships between pages, so that the user can browse from one page to a set of related pages. A fundamental intuition of the work is that Web pages bookmarked and tagged by the user’s close colleagues are more likely to be of significance than those bookmarked and tagged by people unknown to the user. Each user, when bookmarking a page, has the option of sharing to “self,” “team” (i.e., close colleagues), “community” (wider group of colleagues), and “everyone.” This is taken account of when ranking related pages. For example, those shared to “team” by one of the user’s team-members is ranked higher than a page shared by the same person to “community.”

Another approach [43] has proposed creating an ontological structure by combining a purely statistical analysis of folksonomies with a number of additional techniques:

- Terminological resources like WordNet (<http://wordnet.princeton.edu/>) are used, for example, to identify equivalence between tags.
- This is augmented by using Web resources such as Google and Wikipedia. The former is used to suggest alternative spellings, on the basis of the number of occurrences of the various alternatives. Wikipedia can be used to identify new terms which may not occur in conventional dictionaries. Moreover, Wikipedia *URIs* can be regarded as identifiers for many concepts.
- Ontology matching techniques are used, for example, to identify “relationships between tags, between tags and lexical resources, and between tags and elements in existing ontologies.”
- The preceding automatic techniques are enhanced by human intervention, to confirm the results of the automatic techniques, and to obtain information which could not be obtained otherwise.

The Semantic MediaWiki


The Semantic MediaWiki (http://semantic-mediawiki.org/wiki/Semantic_MediaWiki) represents a different approach to combining the power of formal semantics with the ease-of-use associated with Web2.0 [44, 45]. It builds on the success of wikis in enabling collaboration. Specifically, Semantic MediaWiki is a free extension of MediaWiki, the software used by Wikipedia.

Whereas conventional wikis enable users to collaborate to create Web pages, the Semantic MediaWiki enables collaboration to create a knowledge base to complement the Web pages. Conventional wikis have links between pages; a page describing London might contain a sentence “London is the capital of the U.K.” and a link to a page describing U.K. Syntactically this is done by writing `[[U.K.]]`. In the Semantic MediaWiki, the user can explicitly associate a relation with a link; so that the link between the London page and the U.K. page can have the associated relation “is capital of.” This is done by extending the normal wiki syntax and writing `[[is capital of::U.K.]]`.

This is entirely informal, in the sense that the user is free to choose any relation he or she likes, represented by any phrase the user likes. Of course, there is value in people using the same terms, and they can be encouraged to reuse existing relations; it is also possible to define equivalences between different terminologies (e.g., “knows about” can be equated to “is expert in”).

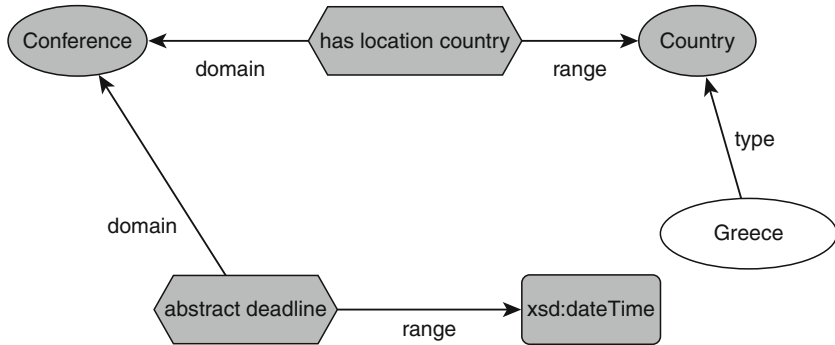
It is possible to use attributes to associate information with a page, other than that which can be represented by relations. For example, the U.K. page could have metadata associated with it describing its population. Syntactically, this can be achieved by writing `[[population: = 61,000,000]]`.

Once a knowledge base has been created using a Semantic MediaWiki, it can then be queried. This can be done using a syntax very similar to the annotation syntax. This is intended for use by the more computer-literate. However, the syntax can be used to create results pages (e.g., a table of the populations of various countries) which can be viewed by everyone. Alternatively, page authors can insert a query enclosed in the `<ask>` tag, so that the displayed page shows not the query but the result of the query.

More recently, an extension to the original Semantic MediaWiki enables forms-based input more suited to end users, see http://www.mediawiki.org/wiki/Extension:Semantic_Forms. Another initiative has generated the capability for nontechnical users of the wiki to create general queries in a relatively easy-to-use way, that is, without using a formal syntax. In this approach, textual queries can be translated into query graphs composed of concepts, relations, and instances in the ontology [46]. In the simplified example quoted in the reference, a user requires to know the deadline for submission to all (presumably forthcoming) conferences in Greece. He or she types the query string “conference Greece deadline.” The resultant query graph is shown in  Fig. 18.15. This is, in effect, a representation of a SPARQL query. The user is then provided with an interface for amending the query graph. He might, for example, wish to change “abstract deadline” to “submission deadline.”

A Lightweight Ontology Editor

The approaches described in [39–41] all in some way draw on the tagging behavior of a user or group of users in order to create or enhance a taxonomy or ontology.



■ Fig. 18.15

Query graph derived from “conference Greece deadline”

The objective is to create a synergy between the formal and informal approaches to knowledge representation. Another way to achieve the same goal is to provide users with an easy-to-use ontology editor, restricted to creating and editing lightweight ontologies. By lightweight ontologies are meant ontologies with relatively limited features, but nevertheless powerful enough for generic knowledge management applications.

Reference [47] describes the use of such an approach to create an ontology editor for the Semantic MediaWiki. The system supports both the import and export of OWL ontologies, and also the import of folksonomies. The latter feature allows a folksonomy dataset to be mapped to an ontology representation. Imported tags are compared with WordNet and Wikipedia, as in [43]. Tags are clustered, mapped to the SKOS knowledge-organization ontology [48] and then mapped and inserted according to the SMW ontology. Additionally, knowledge repair functionalities are provided that assist users with the discovery and mitigation of redundancies and inconsistencies within the knowledge base.

18.2.3 The Semantic Desktop: Supporting the User Throughout His Work

18.2.3.1 Sharing Information and Metadata Across Applications and Desktops

► Section 18.1.5 noted the need for metadata, shared between applications and linked to the context of the user’s work and the processes he or she performs.

One early initiative to address this challenge was the Haystack project [49]. The Haystack project aimed to provide more flexibility in personal information management, and to give the user more control over how information is recorded, annotated, and manipulated. Haystack is now a group at MIT which “develops tools for the web and desktop that can flex to hold and present whatever information a user considers

important, in whatever way the user considers most effective” (<http://groups.csail.mit.edu/haystack/>).

The original version of Haystack preceded RDF, but later RDF was adopted by the project. More recently, the adoption of RDF and semantic technologies has led to an initiative known as the *semantic desktop*. In general terms, the goal of the semantic desktop is to link information objects on the desktop, and on shared servers, through a shared ontology in much the same way as the Semantic Web aims to semantically link objects on the Web. A good description of the early work, along with a number of early references is in [50]. An important aspect of the vision is to allow users to create their own mental models, through the shared ontology. The reference talks about “trails,” which are “paths of resources that build a personal look on a topic.” Another important aspect is the emphasis on a P2P philosophy, so that information objects across desktops are semantically linked. Underlying all this is the need for personal knowledge management tools that can “integrate heterogeneous sources taken from the Semantic Desktop,” which in turn requires ontology mapping techniques.

In Europe, during 2006–2008, the Nepomuk project (<http://nepomuk.semanticdesktop.org>) was a major focus for work on the semantic desktop [51]. Consistent with the previous discussion, the goal of Nepomuk was to link data, and metadata, across applications and across desktops, using shared conceptualizations expressed in RDF. Specifically, the project set out to provide “a standardized description of a Semantic Desktop architecture, independent of any particular operating system or programming language.” A reference implementation of this architecture has been developed, known as Gnowsis. More recently this name has been adopted by a semantic desktop startup, see <http://www.gnowsis.com>.

The project employs an ontology-based approach and uses the *Personal Information Model Ontology* (PIMO) [52], originally developed to represent desktop sources in the EPOS project, which ran from 2003 to 2005 (<http://www3.dfki.uni-kl.de/epos>). Such an ontology allows different applications to share data, while at the same time avoiding the “n:n” problem, that is, the data models for each application map to the PIMO. This is essentially the use of ontologies for data integration, as discussed in [Sect. 18.1.6](#) below. PIMO uses a layered ontology approach, providing generic upper- and mid-level ontologies, and also permitting domain ontologies to be constructed, for example, for a particular company, and leaving users to create ontologies more specific to their needs. This enables users to create their own mental models building on a preexisting base. It avoids the so-called cold start problem where a lack of initial content deters use of the system and the construction of further content. Because the creators of PIMO did not believe that rules or description logic is required for personal information models, modeling is done in RDFS, rather than OWL. The model integrates some third-party ontologies such as the “Friend of a Friend” (FOAF) ontology (<http://www.foaf-project.org/>).

One of the products of Nepomuk was the SPONGE (Semantic Personal Ontology-based Gadget) software tool [53]. The tool “supports users finding, retrieving and annotating desktop resources ... plus seamless access to internet information.” Some

information and interaction is available via a small gadget, taking up limited space on the user's screen. More information is available via the user's browser. The reference claims that future work will extend the functionality with collaborative features. These include the ability to access remote desktops in a P2P topology and workspaces which will facilitate the sharing of resources.

18.2.3.2 Understanding User Context

One of the early goals of the semantic desktop was to understand how the users' information resources divide into a number of contexts, and to detect when a user switches between contexts [49]. This would enable information to be presented to the user, taking account of his or her current context. A number of current projects are investigating this theme.

The APOSDLE project (<http://www.aposdle.tugraz.at/>) is aimed specifically at informal eLearning, that is, at providing the user with small chunks of learning material just when required [54, 55]. This requires understanding the context of the user's current work. For example, in one envisaged scenario the user's actions are analyzed to determine that, for example, he or she is in the starting phase of a project. The user is then provided with information and guidance relevant to project start activity. The project is developing a number of widgets to enable user interaction. These include a context selector; a widget which displays resources relevant to the current context; a global search widget; and a "main" widget which presents the current selected or detected context and possible learning goals. There is also a "cooperation wizard" to guide users through cooperation processes.

APOSDLE is ontology-based. The user creates three types of models: a domain model; a task model describing the tasks which need to be executed; and a learning goal model. Modeling tools are provided, including a semantic wiki and plug-ins for the ontology editor Protégé. The user can also annotate parts of documents using the domain model.

A parallel but separate activity, involving some of the same researchers as in APOSDLE, is also developing a system for task detection [56]. The system is known as UICO, loosely an acronym from "an ontology-based User Interaction COntext model for automatic task detection on the computer desktop." The objective of this work is more general than eLearning, but much of the approach is similar to APOSDLE. An ontology-based user context model has been developed. The model is inspired by the Personal Information Model Ontology discussed above. The ontology, and modeling done in the ontology, form an input to the system's task detection software. To achieve this task detection, the project has developed the concept of the "semantic pyramid." At the bottom layer are events, resulting from "single user interactions with the computer desktop." Above this are event blocks, which are "sequences of events that belong logically together." At the top are tasks, which are "well-defined steps of a process, that cannot be divided into subtasks, and in which only one person is involved." Thus, from the user's viewpoint (rather than the computer's) tasks are essentially atomic. Key to the application of this concept is the delivery of resources relevant to the user's actions.

Another related project is ACTIVE (<http://www.active-project.eu>) [57]. ACTIVE has three main research themes:

- Information delivery guided by user context; this entails the system being able to detect a user's current context.
- The creation of informal processes by users, and the learning of these processes through observation of the user's interaction with his or her computer. By "informal" processes are meant processes designed by individuals to achieve their work-related goals, rather than the formal processes designed on behalf of the organization.
- Knowledge sharing through the synergy of an informal (Web2.0) and an ontology-based approach.

ACTIVE sees context and process as often orthogonal. For example, two of the case studies in the project (e.g., see [58]) are concerned in part with customer-facing people who spend a significant amount of time writing customer proposals. For these users, context will often, but not necessarily, equate to customer. The process, on the other hand, is that of writing a customer proposal, which can be enacted in a number of contexts (i.e., for different customers). As noted above, ACTIVE is seeking to identify both the user's context and his or her current process. The project has developed something similar to the semantic pyramid of UICO. Events as recognized by the machine level need to be combined through various stages to create an understanding of the processes which are intelligible to the users.

ACTIVE aims to impose a minimum of overhead on the user. The user is able to specify his or her set of contexts and to associate information objects with contexts. However, the project is also researching both how to automatically associate information objects with particular contexts and also learn contexts. The latter is a problem in unsupervised learning, that is, how on the basis of the user's actions and the information objects he or she accesses, can those information objects be partitioned into a set of contexts.

Contexts can be shared, that is, a group of users can share the same context; this encourages the sharing of information. Processes can also be shared. This encourages process reuse and also process improvement as colleagues are able to review and improve each others' processes.

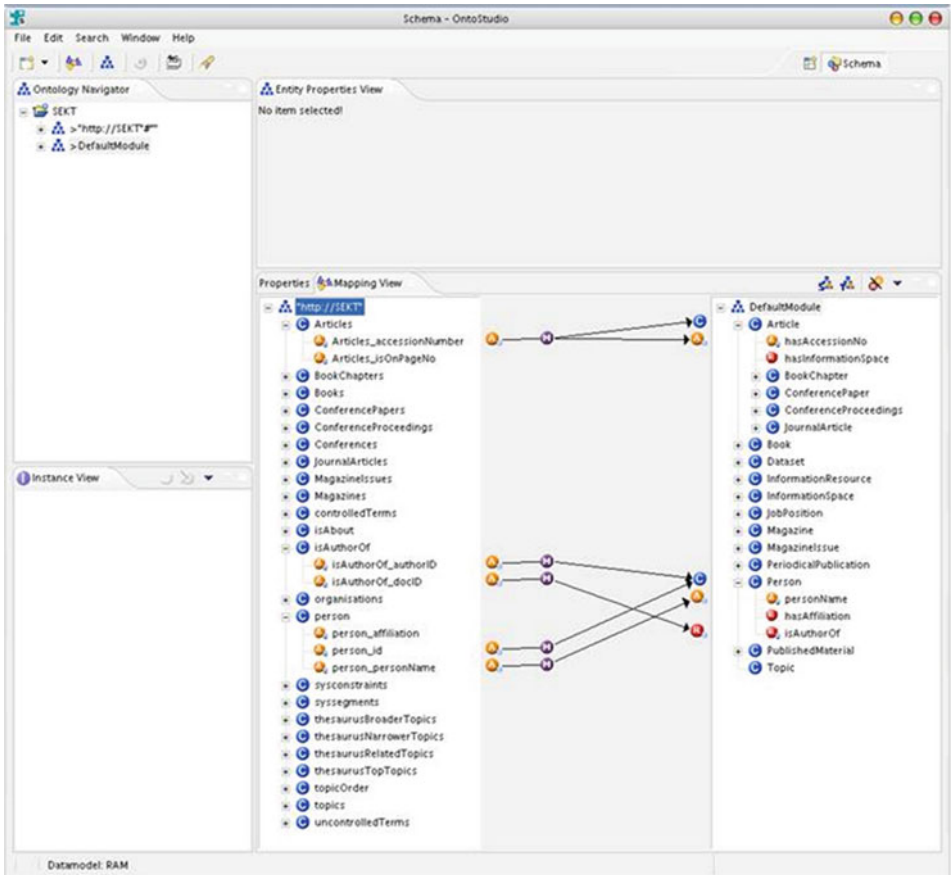
The third theme of ACTIVE is knowledge sharing. This includes continued development of the Semantic MediaWiki and the lightweight ontology editor discussed in [▶ Sect. 18.2.2](#). The goal here is to make use of ontologies in knowledge management, so as for example to be able to exploit reasoning, but in a way which is sufficiently user-friendly for casual, nonspecialist, users.

18.2.4 Graphical and Semiautomatic Approaches to Information Integration

The approach of [▶ Sect. 18.1.6](#) reduces the number of mappings needed, but they still do have to be created. One way to create mappings is to use a mapping language. This is fine

for specialist knowledge engineers but others need a more natural and intuitive approach which is easy to learn and use. A number of graphical mapping tools have been created for such users. One such has been developed by ontoprise GmbH (<http://www.ontoprise.com>) as part of their OntoStudio ontology engineering environment.

Simple drag-and-drop functionality is used to create and amend mappings. At the same time, the system undertakes consistency checks to ensure that the user's actions make sense. ▶ [Figure 18.16](#) shows a view of the mapping tool. The left- and right-hand side shows portions of two different ontologies, and the mappings are represented by lines between them. Mappings can even be conditional. Consider, for example, a mapping between two national transport ontologies. The definition of a “truck” differs in different countries, depending in some countries on the weight of the vehicle. This can be taken into account when creating the mapping.



■ Fig. 18.16

Ontology mapping in OntoStudio. Courtesy: ontoprise GmbH

Even greater gains can be achieved by automating, at least partially, the process of creating the mappings. This is an area of current research. A starting approach is to look for similarities in the text strings used to denote data fields by different schemas, for example, phone for telephone. This can even take account of different representations of similar sounds, for example, the use of “4” to represent “for.” Such an approach is frequently called syntactic matching. Some appreciation of semantics can be introduced by using a thesaurus, such as WordNet, to identify synonyms. Semantic matching can go further by taking account of the structure inherent in two schemas. For example, a product classification system can in general be represented as a graph. Structural similarities then enable the software to draw reasonable conclusions about the relationship between nodes (i.e., categories of products) in two classification systems. The software may propose equivalences between categories, or that a category in one system is a subset of a category in the other classification. Readers interested in the technical detail of one approach, based on the use of a form of logic known as propositional calculus, are referred to reference [59]. For a relatively recent overview of the state of the art in the area of ontology mapping generally, see [60].

Once these techniques have been used to create an initial mapping, it can then be loaded into a graphical editing tool and refined manually.

The end result is that it is possible to integrate heterogeneous databases, and provide the knowledge worker in an organization with a unified view across these databases. This is an important step in reducing the risk of significant information not being available, be it to better inform management decisions or to satisfy regulatory disclosure requirements.

18.2.5 Extracting and Exploiting Semantics from Unstructured Information

▶ [Section 18.1.7](#) identified the need to analyze text so as to create structured knowledge and merge with existing structured knowledge in, for example, relational databases. This section discusses some tools to help achieve this.

18.2.5.1 Software for Text Analytics

▶ [Section 18.1](#) discussed the two approaches to creating metadata; one based on statistics and machine-learning and one based on an analysis of language syntax and grammar known as natural language processing (NLP). The term *text analytics* is used to describe both approaches.

The statistical and machine-learning approach is well represented by the Text-Garden suite of software tools (<http://kt.ijs.si/software/TextGarden/>) developed within the Jozef Stefan Institute in Ljubljana, Slovenia, and used within the SEKT project described earlier [61]. Text mining techniques are also provided as part of the open-source data mining software, Rapid Miner, which is available on SourceForge and supported by Rapid-I GmbH, <http://rapid-i.com>.

The NLP approach is represented by GATE, developed by the University of Sheffield in the U.K., and used in the SEKT project (<http://gate.ac.uk/>); and also by UIMA, originally developed by IBM. An early introduction to GATE is given in [62]; a slightly later, more comprehensive overview is given in [63] GATE is also covered in [▶ Semantic Annotations and Retrieval: Manual, Semiautomatic, and Automatic Generation](#). GATE provides an environment for creating NLP applications. It combines three aspects; it is an architecture, a framework, and a development environment for language engineering. GATE is open and includes a set of resources which others can use and extend. The architecture separates low-level tasks (e.g., data storage, data visualization and location, and loading of components) from data structures and algorithms. The framework provides a reusable design plus software building blocks. The development environment provides tools and a GUI for language engineering. It also provides an interface for text annotation, in order to create training corpora for machine learning algorithms. By an analysis of grammatical structures, such software can, for example, perform named entity recognition and deduce, with reasonable accuracy, to what nouns particular pronouns refer. Such applications are the basis for the semantic search techniques discussed in [▶ Sect. 18.1.3](#) and for the information extraction from text discussed in this section.

UIMA (an acronym for *Unstructured Information Management Applications*) [64] also provides an architecture for the analysis of unstructured text. Having originally been developed by IBM, it is now being developed by the standards body OASIS (<http://www.oasis-open.org>). Apache UIMA is an Apache-licensed open-source implementation of the UIMA specification, see <http://uima.apache.org>. The principle of UIMA is that applications are decomposed into components. The UIMA framework defines the interfaces between these components and manages the components and the data flows between them. As noted in the reference above: “The principal objective of the UIMA specification is to support interoperability among analytics.” This is divided into four design goals:

- Data representation – supporting the common representation of artifacts and metadata
- Data modeling and interchange – supporting the platform-independent interchange of artifacts and metadata
- Discovery, reuse, and composition of independently developed analytics tools
- Service-level interoperability – supporting the interoperability of independently developed analytics based on a common service description and associated SOAP bindings

GATE and UIMA are overlapping in scope and an interoperability layer has now been created between them; one view sees GATE’s advantage as a prototyping tool while UIMA’s advantages are in performance and scalability [65].

18.2.5.2 Extracting Information from the World Wide Web

The previous discussion has assumed that the information to be integrated resides within an enterprise. The rise of the World Wide Web has provided one motivation for

combining data from outside the enterprise. An approach to achieving this is described in [66]. Here an ontology is used to provide a view across information on the Web. In the future world of the Semantic Web, much information on the Web will be described using ontologies, and the problem will be to map from these into an overarching one. Today data on the Web exist in variety of forms, for example, unstructured or semi-structured HTML files. The first step is frequently to extract the desired data and to describe them in terms of the ontology. The next step is to undertake instance matching, that is, to identify equivalent instances. The paper proposes a scalable approach based on the use of a group of peers. However, more relevant to the interests of this handbook is the use of similarity metrics to construct the mappings between instances. The authors investigated three sets of features to characterize similarity: character level, word level, and ontological level. The first of these is determined by the number of character transformations to edit from one string to another (the so-called Levenshtein distance [67]) and the second is based on the “bag of words” approach common in information retrieval. The ontological similarity attempts to measure the distance between two concepts. For example, at the extremes, if two concepts are the same the distance is 0, while if they are disjoint the distance is infinite (represented in practice by a very large positive number). If two instances are known to instantiate two concepts, then intuitively the larger the concept distance, the smaller the probability of these instances being the same. The paper reports an experiment in which a method incorporating all three approaches had higher precision than other methods at “almost” all recall levels; although the higher the recall the less advantageous the incorporation of the ontological approach.

18.2.6 Sharing Information Across Organizations

▶ Section 18.1.8 talked about the need for shared vocabularies where organizations need to collaborate, and noted the problems which arise because such vocabularies are frequently informally defined. Two approaches to sharing data were noted. On the one hand, within a given domain, existing informal vocabularies can be formalized. This is the approach discussed in ▶ Sect. 18.2.6.1, where medicine is taken as an example.

On the other hand, where one is starting from scratch, self-describing datasets can be made available on the Web, and linked as appropriate. Where these datasets are made openly available, this creates a Web of linked open data. This approach is described very briefly in ▶ Sect. 18.2.6.2; much more detail on this topic is provided in ▶ [Semantic Annotation and Retrieval: Web of Data](#), this volume.

18.2.6.1 An Example from Medicine

In medicine and biology, the vocabularies are often very large and complex. This was a natural area, therefore, for the early application of ontologies. In fact, the most well known of all ontology tool suites, Protégé (<http://protege.stanford.edu/>), was originally

motivated by the needs of medical informatics, and this domain continues to influence its development. Today, there are a very large number of biomedical ontologies. Reference [68] gives a brief introduction, making the case for the development of virtual ontology repositories which could be browsed by potential users looking for an appropriate ontology, prior to downloading.

In the area of clinical medicine, the best-known example of a shared ontology is SNOMED-CT (Systematized **N**omenclature of **M**EDicine – **C**linical **T**erms). This was created, in 2002, by the merger of SNOMED-RT (Reference Terminology) from the College of American Pathologists and the UK National Health Service Clinical Terms. It is now maintained by the International Health Terminology Standards Development Organization (IHTSDO, see: <http://www.ihtsdo.org/>). SNOMED-CT is a very large vocabulary; by August 2008 it had 283,000 concepts.

SNOMED-CT was not originally designed as an ontology. However, as ontologies were being discussed in knowledge management, medical informatics was an obvious candidate for their application. Reference [69] is an early paper discussing how ontologies could be relevant to medical vocabularies such as SNOMED. The paper saw ontologies being applied in medicine to areas such as natural language processing, that is, to conceptualize language and serve an “interlingual” role; and supporting simulation and modeling, for example, in molecular biology; and knowledge sharing. They also note the difficulty that medical concepts are empirical rather than being perfectly defined. All this, of course, applies to many other specialist areas. The authors also lay down some principles for creating well-formed ontologies; this again is applicable to any domain area, not just medicine.

Despite not being originally conceived as an ontology, SNOMED adopted description logic as its representation language. Moreover, since the development of OWL1.1 it has been possible in principle to translate SNOMED into OWL. A discussion of what is involved in this is given in [70]; the barriers to achieving this are largely due to the size of SNOMED.

One valuable feature of the description logic approach is that of “post-coordination.” Whilst as already noted, SNOMED has a very large number of defined concepts, post-coordination helps reduce the number required. Post-coordination means that new concepts can be created from preexisting concepts, for example, by a clinician. Automatic consistency checking is required at the time the new concept is created.

18.2.6.2 The Web of Linked Data

The WWW as it has initially evolved is a Web of interlinked documents. Berners-Lee’s original vision, however, went beyond this to a parallel Web of Data. That this vision was not realized at the same time as the Web of Documents was probably due, at least in part, to the lack of finalized standards to describe data in the early years of the millennium, for example, the RDF standard was not finalized until 2004. However, in 2006 Berners-Lee returned to the subject of the Web of Data by publishing a set of principles for linked data [71].

The four principles which Berners-Lee enunciated are:

1. Use URIs as names for things.
2. Use HTTP URIs so that people can look up those names.
3. When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL).
4. Include links to other URIs so that they can discover more things.

The first two of these are familiar as a foundation for the Semantic Web. The third means that data will be self-describing. Finally, the fourth is a basis for the WWW, or indeed for any Web; through interconnectivity crawlers can discover all the data available.

The result is now called the Web of Linked Open Data, and is the subject of a W3C taskforce (<http://esw.w3.org/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>). By July 2009, the Web of linked open data contained 6.7 billion triples and 149 million links [72].

As already noted, [Semantic Annotation and Retrieval: Web of Data](#) provides detailed information about the Web of Data, including descriptions of application areas; linked open data sources; and some of the available tools, such as Web of Data search engines.

Linked open data is provided by organizations who want to make information publicly available, for example, because they are government organizations with a mandate to do so, or because the information is about products which they market. Many of the consumers of such data will in turn make data openly available themselves. However, there are two broad scenarios where commercial organizations can make use of linked open data: firstly, to link internal and external data sources to add value to internal “own-use” applications and secondly, to build applications for customers based on public data. In the first category, one can imagine, for example, the use of public demographic data to enhance targeted marketing applications. In the latter category, an example would be to offer personalized location-based services by accessing public data about a particular location.”

The principles of linked open data and the technology developed for linked open data could equally well be used within organizations, to create data intranets, or between organizations to create data extranets. Again by analogy to the Web of Documents, links from these data intranets and extranets could reach out to the open data web, while links in the reverse direction would not, of course, be traversable.

18.3 Related Resources

An extensive list of references is given in the reference section. This section offers a non-exhaustive list of some of the key resources on the application of semantic technology to knowledge management.

“*Ontologies for knowledge management,*” Abecker, A., & van Elst, L. in “*Handbook on Ontologies,*” Studer, R. & Staab, S. (eds), Springer-Verlag, 2003.

This book chapter offers an excellent brief survey of the role ontologies can play in knowledge management systems. KM and the requirements on IT systems are introduced. The areas where ontologies can play a part in meeting those requirements are then discussed. An analysis of future practice and research, and the outlook for future trends and developments in ontology-based KM systems are given.

“*Towards the Semantic Web: Ontology-driven Knowledge Management*,” Davies, J., Fensel, D. & van Harmelen, F. (eds), Wiley, Chichester, UK, 2003.

Based on results from the OnToKnowledge project (one of the first European research projects looking at the relationship between semantic technology and knowledge management), this book covers basic research, tools, and case studies in ontology-driven knowledge management and offers a good overview of early work on this topic.

“*Ontologies for Knowledge Management: An Information Systems Perspective*,” Jurisica, I., Mylopoulos, J., Yu, E., *Knowledge and Information Systems*, Vol 6, No 4, Springer, London, 2004.

This paper surveys approaches to knowledge representation in Computer Science and categorizes them into four ontological categories: static ontologies, dynamic ontologies, intentional ontologies, and social ontologies. The benefits and drawbacks of the ontological approach are also discussed and the use of ontologies motivated at a foundational level.

“*Information Integration with Ontologies*,” Alexiev, A., Breu, M., de Bruijn, J., Fensel, D., Lara, R. & Lausen, H., Wiley, Chichester, UK, 2005.

This book describes how ontology technology can be used to manage dispersed, heterogeneous information assets more efficiently. The book compares the ontological approach with current EAI technology. One strength of the book is that examples are taken from an industrial application using real data sources from the automotive sector.

“*Semantic Knowledge Management: Integrating Ontology Management, Knowledge Discovery, and Human Language Technologies*,” Davies, J., Grobelnik, M. & Mladenic, D., Springer, Berlin, 2009.

This book presents a framework, methods, and tools for semantic knowledge management, which it defines as the use of semantic technology for improved management of tangible knowledge assets. An interdisciplinary approach is advocated and discussed involving the use of knowledge discovery, ontology management, and human language technologies. Applications using the underlying technologies are described, along with a series of evaluated case studies showing the value of the semantic approach to KM in real-world settings.

18.3.1 Semantic Web Interest Group: Case Studies and Use Cases

The Semantic Web Interest Group (<http://www.w3.org/2001/sw/sweo/>), which has now closed, has produced a wide range of case studies and use cases, see <http://www.w3.org/2001/sw/sweo/public/UseCases/>. Here case studies refers to deployed systems while use cases refers to prototypes. They can be sorted along a number of dimensions, including application area and technologies used.

18.4 Future Issues

18.4.1 Web2.0 and Ontologies

The success of the informal Web2.0 techniques, discussed in [Sect. 18.2.2.1](#), is a challenge to semantic technologies. Can this success be further strengthened by combining these techniques with more formal techniques? This has been discussed in some depth. However, significant challenges remain.

The creation and use of ontologies needs to be simple and intuitive. It needs to be recognized that there are different constituencies to be catered for. There are some users who should not be aware of the existence of an ontology (or even what the word means), but who need straightforward tagging features, with software automatically creating and using an underlying ontology. At the other extreme there are power users, perhaps professionals in biomedical research, who will want to interact directly with the full power of ontologies – although even for them all interfaces should be as simple as possible, nothing should be more complex than it needs to be. There may be grades of users in between, requiring to understand something about ontologies and interact directly with them; although the language of ontologies may be too off-putting and other terminology may be more appropriate. There will also be people akin to database administrators who will create and maintain ontologies. Again, there will be a range of such people, depending in part on the nature of the applications. Some will have little formal training in IT; others will be IT professionals. The tools offered need to reflect this.

As far as is possible, the creation and maintenance of ontologies needs to be automatic. This requires the use of techniques from information retrieval (e.g., based on the “bag of words” approach) and natural language processing. There may be scope for combining these two approaches to provide increased user functionality. There are also user interface issues here. For example, there is a need to understand to what extent the process of metadata creation can be entirely automated and to what extent the user needs to confirm suggestions; and how this can be done in an unobtrusive way.

18.4.2 Integrating into and across Enterprises

McKinsey claim that “successful companies not only tightly integrate Web2.0 technologies with the workflows of their employees but also create a “networked company,” linking themselves with customers and suppliers through the use of Web2.0 tools” [38]. This highlights two challenges for applying semantic technology in the enterprise.

Firstly, there is a need to refine technologies such as those of the semantic desktop discussed in [Sect. 18.2.3](#) which integrate metadata across applications and with workflows. Integration of metadata with informal workflows created by information

system users, not just the formal ones created by the organization, is important to enable tools for improved productivity.

Secondly, building on the use of semantic technology to overcome heterogeneity within the organization, there is a need to use these technologies to address the even greater heterogeneity which exists when organizations work closely together. This may be in a supply chain or with customers; it may be for a relatively long period, or it may require that a collaboration infrastructure be created rapidly, used for a few months, and then withdrawn. Improved ontology mapping techniques will be required. As with the generation of automatic metadata, the need is to understand how to combine automatic and manual mapping techniques; and how to do this in a way which is natural for users who may not be IT professionals. Uschold and Gruninger [28] propose a methodology for making progress in research into achieving interconnectivity. They believe that working systems will require many assumptions and that research progress will be made by relaxing these assumptions one by one. Examples of such assumptions include use of a single ontology language, use of a single shared ontology, or use of a single shared upper ontology with distinct domain ontologies.

18.5 Cross-References

- eBusiness
- eGovernment
- eScience
- Future Trends
- Multimedia, Broadcasting and eCulture
- Ontologies and the Semantic Web
- Semantic Web Search Engines
- Social Semantic Web

References

1. Drucker, P.: Knowledge-worker productivity: the biggest challenge. *Calif. Manag. Rev.* **41**, 79–94 (1999)
2. Ackoff, R.L.: From data to wisdom. *J. Appl. Syst. Anal.* **16**, 3–9 (1989)
3. Chadran, A.: Economist Intelligence Unit: enterprise knowledge workers: understanding risks and opportunities. www.eiu.com/knowledgeworkers (2007). Accessed 29 Dec 2010
4. Jansen, B.J., Spink, A., Saracevic, T.: Real life, real users, and real needs: a study and analysis of user queries on the web. *Inf. Process. Manag.* **36**(2), 207–227 (2000)
5. Davies, J., Kiryakov, A., Duke, A.: Semantic search. In: Goker, A., Davies, J. (eds.) *Information Retrieval: Searching in the 21st Century*. Wiley, London (2009)
6. Russell-Rose, T., Stevenson, M.: The role of natural language processing in information retrieval. In: Goker, A., Davies, J. (eds.) *Information Retrieval: Searching in the 21st Century*. Wiley, London (2009)

7. Dumais, S., Cutrell, E., Cadiz, J., Jancke, G., Sarin, R., Robbins, D.: Stuff I've seen: a system for personal information retrieval and re-use. In: Proceedings of 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003), Toronto. ACM Press, New York (2003)
8. Kiryakov, A.: Ontologies for knowledge management. In: Davies, J., Studer, R., Warren, P. (eds.) *Semantic Web Technologies: Trends and Research in Ontology-Based Systems*. Wiley, Chichester (2006)
9. Jones, W.: Finders, keepers? The present and future perfect in support of personal information management. *First Monday* 9(3) (2004). http://131.193.153.231/www/issues/issue9_3/jones/index.html
10. Jones, W.: *Keeping Found Things Found*. Morgan Kaufmann, San Francisco (2007)
11. Civan, A., Jones, W., Klasnja, P., Bruce, H.: Better to organise personal information by folders or by tags? The devil is in the details. In: Paper Presented at the 68th Annual Meeting of the American Society for Information Science and Technology (ASIS&T 2008). Columbus. <http://www.asis.org/> (2008)
12. Davenport, T.H.: Knowledge management case study; knowledge management at Ernst & Young. Information technology management white paper. <http://www.itmweb.com/essay537.htm> (1997). Accessed 29 Dec 2010
13. Ezingard, J., Leigh, S., Chandler-Wilde, R.: Knowledge management at Ernst & Young UK: getting value through knowledge flows. In: Proceedings of the 21st International Conference on Information Systems (ICIS 2000), Brisbane, pp. 807–822 (2000)
14. McComb, D.: *Semantics in Business Systems: The Savvy Manager's Guide*, Chapter 12. Morgan Kaufmann, San Francisco (2004)
15. Pollock, J., Hodgson, R.: *Adaptive Information: Improving Business Through Semantic Interoperability, Grid Computing, and Enterprise Integration*. Wiley-Interscience, Hoboken (2004)
16. Bernstein, P., Haas, L.: Information integration in the enterprise. *Commun. ACM* 51(9), 72–79 (2008)
17. Halevy, A., Ashish, N., Bitton, D., Carey, M., Draper, D., Pollock, J., Rosenthal, A., Sikka, V.: Enterprise information integration: successes, challenges and controversies. In: Proceedings of the ACM SIGMOD 2005 International Conference on Management of Data, Baltimore, pp. 778–787. ACM Press, New York (2005)
18. World Wide Web Consortium Semantic web use cases and case studies. <http://www.w3.org/2001/sw/sweo/public/UseCases/>. Accessed 29 Dec 2010
19. Doan, A., Noy, N., Halevy, A.: Introduction to the special issue on semantic integration. *SIGMOD Rec.* 33(4), 11–13 (2004)
20. Noy, N.: Semantic integration: a survey of ontology-based approaches. *SIGMOD Rec.* 33(4), 65–70 (2004)
21. Bernstein, P., Melnik, S., Petropoulos, M., Quix, C.: Industrial-strength schema matching. *SIGMOD Rec.* 33(4), 38–43 (2004)
22. Bernstein, P., Melnik, S.: Model management 2.0: manipulating richer mappings. In: Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data (SIGMOD/PODS 2007), Beijing, pp. 1–12 (2007)
23. Moore, C. (vice president and research director Forrester Research): Information indepth, Oracle. <http://www.oracle.com/newsletters/information-insight/content-management/feb-07/index.html> (2007). Accessed 29 Dec 2010
24. Murphy, B., Markham, R.: *eDiscovery Bursts Onto the Scene*. Forrester, Cambridge (2006)
25. AIIM: Electronic communication policies and procedures. A 2005 industry study prepared jointly by AIIM and Kahn Consulting Inc. <http://www.aiim.org> (2005). Accessed 29 Dec 2010
26. Foxvog, D., Bussler, C.: Ontologizing EDI: first steps and initial experience. In: Proceedings of the International Workshop on Data Engineering Issues in E-Commerce (DEEC 2005), Tokyo, pp. 49–58. IEEE Computer Society, Los Alamitos (2005)
27. Abecker, A., van Elst, L.: Ontologies for knowledge management. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*, Chapter 22, pp. 435–454. Springer, Dordrecht (2004)
28. Uschold, M., Gruninger, M.: Ontologies and semantics for seamless connectivity. *SIGMOD Rec.* 33(4), 58–64 (2004)
29. Duke, A., Heizmann, J.: Semantically enhanced search and browse. In: Davies, J., Grobelnik, M., Mladenic, D. (eds.) *Semantic Knowledge Management*, pp. 85–102. Springer, Berlin (2009)

30. Bontcheva, K., Davies, J., Duke, A., Glover, T., Kings, N., Thurlow, I.: Semantic information access. In: Davies, J., Studer, R., Warren, P. (eds.) *Semantic Web Technologies: Trends and Research in Ontology-Based Systems*, pp. 139–169. Wiley, Chichester (2006)
31. Motik, B., Studer, R.: KAON2: a scalable reasoning tool for the semantic web. In: *Proceedings of the Second European Semantic Web Conference (ESWC 2005)*, Heraklion (2005)
32. Thurlow, I., Warren, P.: Deploying and evaluating semantic technologies in a digital library. In: Davies, J., Grobelnik, M., Mladenic, D. (eds.) *Semantic Knowledge Management*, pp. 181–198. Springer, Berlin (2009)
33. Bloehdorn, S., Görlitz, O., Schenk, S., Völkel, M.: TagFS – tag semantics for hierarchical file systems. In: *Proceedings of the Sixth International Conference on Knowledge Management (I-KNOW 2006)*, Graz (2006)
34. Franz, J., Traphöner, R.: Semantic Web for knowledge reuse in business processes. In: Davies, J., Grobelnik, M., Mladenic, D. (eds.) *Semantic Knowledge Management*, pp. 215–229. Springer, Berlin (2009)
35. Warren, P., Thurlow, I., Alsmeyer, A.: Applying semantic technology to a digital library. In: Davies, J., Studer, R., Warren, P. (eds.) *Semantic Web Technologies: Trends and Research in Ontology-Based Systems*, pp. 237–257. Wiley, Chichester (2006)
36. Bontcheva, K., Cunningham, H., Kiryakov, A., Tablan, V.: Semantic annotation and human language technology. In: Davies, J., Studer, R., Warren, P. (eds.) *Semantic Web Technologies: Trends and Research in Ontology-Based Systems*, pp. 29–50. Wiley, Chichester (2006)
37. Sure, Y., Tempich, C., Vrandečić, D.: Ontology engineering methodologies. In: Davies, J., Studer, R., Warren, P. (eds.) *Semantic Web Technologies: Trends and Research in Ontology-Based Systems*. Wiley, Chichester (2006)
38. Bughin, J., Chui, M., Miller, A.: How companies are benefiting from web 2.0. McKinsey (2009). <https://www.mckinseyquarterly.com/>. Accessed 29 Dec 2010
39. Millen, D., Feinberg, J., Kerr, B.: Social bookmarking in the enterprise. *ACM Queue* 3(9), 28–35 (2005). <http://researchweb.watson.ibm.com/jam/601/p28-millen.pdf>
40. Hayman, S.: Folksonomies and tagging: new developments in social bookmarking. In: *Proceedings of the Ark Group Conference: Developing and Improving Classification Schemes*, Sydney. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.138.8884&rep=rep1&type=pdf> (2007)
41. Heymann, P., Garcia-Molina, H.: Collaborative creation of communal hierarchical taxonomies in social tagging systems. Technical report 2006-10, Stanford University. <http://ilpubs.stanford.edu:8090/775/> (2006)
42. Kings, N., Gale, C., Davies, J.: Knowledge sharing on the semantic web. In: Franconi, E., Kifer, M., May, W. (eds.) *Proceedings of the Fourth European Semantic Web Conference (ESWC 2007)*, Innsbruck. *Lecture Notes in Computer Science*, vol. 4519, pp. 281–295. Springer, Berlin (2007)
43. Van Damme, C., Hepp, M., Siorpaes, K.: FolksOntology: an integrated approach for turning folksonomies into ontologies. In: *Proceedings of the Fourth International European Semantic Web Conference (ESWC 2007) – Bridging the Gap Between Semantic Web and Web 2.0*, Innsbruck. *Lecture Notes in Computer Science*, vol. 4519. Springer, Berlin. <http://www.kde.cs.uni-kassel.de/ws/eswc2007/proc/FolksOntology.pdf> (2007)
44. Krötzsch, M., Vrandečić, D., Völkel, M.: Semantic MediaWiki. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) *Proceedings of the Fifth International Semantic Web Conference (ISWC 2006)*, Athens, GA. *Lecture Notes in Computer Science*, vol. 4273, pp. 935–942. Springer, Berlin/Heidelberg (2006)
45. Vrandečić, D., Krötzsch, M.: Semantic MediaWiki. In: Davies, J., Grobelnik, M., Mladenic, D. (eds.) *Semantic Knowledge Management*, pp. 171–179. Springer, Berlin (2009)
46. Haase, P., Herzig, D., Musen, M., Tran, T.: Semantic wiki search. In: *Proceedings of the Sixth European Semantic Web Conference (ESWC 2009)*, Heraklion. *Lecture Notes in Computer Science*, vol. 5554, pp. 445–460. Springer, Berlin (2009)
47. Luger, M., Wölger, S., Bürger, T.: SMW ontology editor – features. Internal report. STI Innsbruck, University of Innsbruck
48. World Wide Web Consortium SKOS simple knowledge organisation system. <http://www.w3.org/2004/02/skos/>. Accessed 29 Dec 2010

49. Karger, D.R.: Haystack: per-user information environments based on semistructured data. In: Kaptelinin, V., Czerwinski, M. (eds.) *Beyond the Desktop Metaphor: Designing Integrated Digital Work Environments*. MIT Press, Cambridge (2007)
50. Sauer mann, L., Bernardi, A., Dengel, A.: Overview and outlook on the semantic desktop. In: *Proceedings of the First Workshop on the Semantic Desktop (SemDesk 2005)*, International Semantic Web Conference (ISWC 2005), Galway (2005)
51. Groza, T., Handschuh, S., Moeller, K., Grimnes, G., Sauer mann, L., Minack, E., Mesnage, C., Jazayeri, M., Reif, G., Gudjonsdottir, R.: The NEPOMUK project – on the way to the social semantic desktop. In: *Proceedings of the Third International Conference on Semantic Technologies (I-Semantics 2007)*, Graz, pp. 201–211 (2007)
52. Sauer mann, L., van Elst, L., Dengel, A.: PIMO – a framework for representing personal information models. In: *Proceedings of the Third International Conference on Semantic Technologies (I-Semantics 2007)*, JUCS, Graz, pp. 270–277 (2007)
53. Papailiou, N., Christidis, C., Apostolou, D., Mentzas, G., Gudjonsdottir, R.: Personal and group knowledge management with the social semantic desktop. In: Cunningham, P., Cunningham, M. (eds.) *Collaboration and the Knowledge Economy: Issues, Applications and Case Studies*. IOS Press, Amsterdam (2008). ISBN 978-1-58603-924-0
54. Lindstaedt, S., Mayer, H.: A storyboard of the APOSDLE vision. In: *Poster submitted to the First European Conference on Technology Enhanced Learning (EC-TEL 2006)*, Crete (2006)
55. Musielak, M., Hambach, S., Christl, C.: APOSDLE contextualized cooperation. In: *ACM SIGCHI: ACM Conference on Computer Supported Cooperative Work 2008*. Electronic Proceedings: CSCW 08 [CD-ROM], San Diego. ACM Press, New York (2008)
56. Rath, A., Devaurs, D., Lindstaedt, S.: UICO: An ontology-based user interaction context model for automatic task detection on the computer desktop. In: *Proceedings of the First Workshop on Context, Information and Ontologies (CIAO 2009)*. ACM International Conference Proceedings Series, Heraklion (2009)
57. Warren, P., Kings, N., Thurlow, I., Davies, J., Bürger, T., Simperl, E., Ruiz, C., Gómez-Pérez, J., Ermolayev, V., Ghani, R., Tilly, M., Bösser, T., Imtiaz, A.: Improving knowledge worker productivity – the ACTIVE integrated approach. *BT Technol. J.* **26**(2), 165–176 (2009)
58. Warren, P., Thurlow, I., Kings, N., Davies, J.: Knowledge management at the customer front-line – an integrated approach. *J. Inst. Telecommun. Prof.* **3**(4), 8–15 (2009)
59. Bouquet P., Serafini L., Zanobini S.: Semantic coordination: a new approach and an application. In: *Proceedings of the Second International Semantic Web Conference (ISWC 2003)*, Sanibel Islands. Lecture Notes in Computer Science, vol. 2870, pp. 130–145. Springer, Berlin. <http://citeseer.ist.psu.edu/bouquet03semantic.html> (2003)
60. Euzenat, J., Shvaiko, P.: *Ontology Matching*. Springer, New York (2007). ISBN 3540496114
61. Mladenic, D.: Text mining in action! In: *From Data and Information Analysis to Knowledge Engineering: Proceedings of the 29th Annual Conference of the Gesellschaft für Klassifikation e.V., University of Magdeburg* (2005)
62. Cunningham, H., Maynard, D., Bontcheva, K., and Tablan, V.: GATE: an architecture for development of robust HLT. In: *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL 2002)*, Philadelphia, pp. 168–175 (2002)
63. Bontcheva, K., Tablan, V., Maynard, D., Cunningham, H.: Evolving GATE to meet new challenges in language engineering. *Nat. Lang. Eng.* **10**(3–4), 349–373 (2004)
64. OASIS: *Unstructured Information Management Architecture (UIMA) version 1.0*, Working Draft 05 (2008)
65. Roberts, I.: GATE and IBM’s UIMA – interoperability layer. http://videlectures.net/gate06_roberts_giul/ (2006)
66. Wang, C., Lu, J., Zhang, G.: An ontology data matching method for web information integration. In: *Proceedings of the Tenth International Conference on Information Integration and Web-based Applications & Services (iiWAS 2008)*, Linz (2008)
67. В.И. Левенштейн (1965) Двоичные коды с исправлением выпадений, вставок и замещений символов. Доклады Академий

- Hayk CCCP 163.4:845–848. Appeared in English as: Levenshtein, V. I.: Binary codes capable of correcting deletions, insertions, and reversals. *Sov. Phys. Doklady*. **10**, 707–710 (1966)
68. Noy, N., Rubin, D., Musen, M.: Making biomedical ontologies and ontology repositories work. *IEEE Intell. Syst.* **19**(6), 78–81 (2004)
69. Burgun, A., Botti, G., Fieschi, M., Le Beux, P.: Sharing knowledge in medicine: semantic and ontologic facets of medical concepts. In: *Proceedings of IEEE International Conference on Systems, Man and Cybernetics (IEEE SMC 1999)*, Tokyo, pp. 300–305 (1999)
70. Spackman, K.: An examination of OWL and the requirements of a large health care terminology. In: *Proceedings of the Third OWL: Experiences and Directions Workshop (OWLED 2007)*, CEURWS, Innsbruck (2007)
71. Berners-Lee, T.: Linked data – design issues. <http://www.w3.org/DesignIssues/LinkedData.html> (2006). Accessed 29 Dec 2010
72. Bizer, C.: The emerging web of link data. *IEEE Intell. Syst.* **24**(5), 87–92 (2009)
73. Davies, J., Studer, R., Sure, Y., and Warren, P.: Next generation knowledge management. *BT Technol. J.* **23**(3), 175–190 (2005)
74. Cimiano, P., Volker, J.: Text2Onto - A framework for ontology learning and data-driven change discovery. In: *Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB 2005)*, Alicante (2005)

19 eBusiness

Christoph Grün · Christian Huemer · Philipp Liegl · Dieter Mayrhofer · Thomas Motal · Rainer Schuster · Hannes Werthner · Marco Zapletal
Vienna University of Technology, Vienna, Austria

19.1	<i>Scientific and Technical Overview</i>	789
19.1.1	Different Definitions	790
19.1.2	A Short History and the Current Situation	792
19.1.3	eCommerce Applications and Services	795
19.2	<i>Application Domain: Business-to-Business</i>	796
19.2.1	Business Models	798
19.2.1.1	e3-value	799
19.2.1.2	Resource Event Agents (REA)	801
19.2.1.3	The Business Model Ontology (BMO)	803
19.2.2	Business Process Models	804
19.2.2.1	Notation Approaches	804
19.2.2.2	Semantic Approaches	808
19.2.3	Business Documents	816
19.2.3.1	Business Document Standards	816
19.2.3.2	Product Data Catalogs	820
19.3	<i>Application Domain: Business-to-Consumer</i>	822
19.3.1	The Tourism Industry and Its Characteristics	823
19.3.2	The Tourist Life Cycle	824
19.3.3	Trip Planning: Relevant Issues	825
19.3.3.1	Trip Planning Applications	826
19.4	<i>Related Resources</i>	834
19.4.1	Key Articles	834
19.4.1.1	Business Models	834
19.4.1.2	Business Process Models	834
19.4.1.3	Business Document Models	836
19.4.1.4	eTourism	837

19.4.2	Relevant Ontologies	838
19.4.3	Key Events	839
19.5	<i>Future Issues</i>	841
19.6	<i>Cross-References</i>	842

Abstract: Integrating Semantic Web concepts into the domain of eBusiness is a hot topic. However, most of the efforts spent so far concentrated on the improvement on B2C (business-to-consumer) eCommerce applications, achieved by the semantic enrichment of information. With the growing importance of Service-Oriented Architectures (SOA) companies started to move into the section of the Electronic Data Interchange (EDI), where applications exchange their business information semiautomatically. This B2B (business-to-business) electronic commerce is driven by aligning the internal business processes of companies to publicly available business processes. Thereby companies often do not consider the economic drivers of their business processes, which leads to incompatibilities between management, administration, and technical layers. This chapter covers the two major domains of eBusiness/eCommerce, namely B2B and B2C. In the first, a model-driven approach toward B2B IT solutions is introduced, covering semantic aspects dealing with business models, business process models, and business document models. In the second application domain, the basic concepts of Semantic Web in the area of B2C eCommerce are examined using a representative example from the eTourism domain.

19.1 Scientific and Technical Overview

Information Technology has changed and penetrated one's life, business, and society already so much that even visions cannot exist without technology and its applications. And this process is accelerating. The Web and the related eCommerce/eBusiness phenomena (as Web-based eCommerce) is just one example of this development. Although initial (at the end 1990s) eCommerce hypes – especially at the stock markets – have not been fulfilled, eCommerce and eBusiness have changed dramatically the way business is done. In this context one may observe a metamorphosis from the computer to a media machine, based on a global infrastructure (Internet/Web), with a transparent technology and access. This change is based on an evolution of the computer from an automaton (focus is on the manipulation of well-formalized and mathematical models) to a tool (modeling of work processes) and finally a “medium” (with representation and processing of unstructured information).

It is worth mentioning that these developments, leading to structural as well as behavioral changes in society, have already been foreseen more than 40 years ago by authors such as Norbert Wiener, Daniel Bell, or Peter Drucker, referring to the notion of Information Society. They discussed issues such as the appearance of new technologies and economic sectors as well as their convergence. In this context an ever-increasing flexibility and division of labor with a move to the so-called service industry can be observed.

Two major phenomena of today, constituting the context of eCommerce, can be highlighted: acceleration and complexity [96]: a rapidly evolving technological progress is witnessed, steadily shrinking time intervals between the introduction of new inventions and innovative products. But acceleration is a historic phenomena: taking the three major

technologies of mankind – hunting, farming, and industry – each one has grown 100 times faster than its predecessor [30]. This is paralleled by the growth of the so-called knowledge-based industries. Knowledge, acquired through investments in research and development (R&D) as well as education, is recognized as a critical factor for innovation and source of competition. As a matter of fact, international R&D spending has grown over the last decades and R&D-oriented companies have shown a strong performance.

The second phenomenon is complexity, which can also be seen in such a development as globalization, paralleled by an ongoing differentiation. When trying to identify a single aspect of this society by using a social, economic, ecological, or even a cultural point of view, one realizes an ongoing trend toward organization with a simultaneous growth of interdependencies. There exists a relationship between the growth of organizations and complexity. Large organizations are also large information processing systems. The ability to digest information is one of the preconditions for their functioning. In fact, the work of most of them is predominantly in information processing.

The increasing complexity can also be observed on the level of market structures, where at the same time disintermediation and re-intermediation, and enabled by a worldwide IT infrastructure, permanently entering new companies can be seen. Currently, there exists a networked economy, integrating all market participants and consumers. Especially consumers become more active, leading to the phenomena of “prosumption.” It is not sufficient to be customer focused, instead companies and markets are driven by customers.

The complexity of today’s society is correlated with the information processing machinery, which, however, produces also an overabundance of information. In that sense IT-based information processing increases complexity. There is an obvious paradox: on the one hand IT increases the amount of information and complexity, and on the other hand it appears to be the only means to reduce uncertainty, but implying again more IT applications.

19.1.1 Different Definitions

Given the still growing importance of eCommerce and eBusiness, it is interesting to note that there are varying definitions (and statistics). Computer scientists normally refer to the technical issues and building blocks – understanding eCommerce as applied Computer Science – whereas the management science or information system community follows a business and transaction view. And there are broad and narrow definitions: either distinguishing between eBusiness and eCommerce (seeing the latter as part of the first) or not, in this case both terms are (nearly) interchangeable (see the discussion published in [5]).

On one side you could position eCommerce as “is sharing business information, maintaining business relationships, and conducting business transactions by means of telecommunication networks” [36] with the focus on the coverage of all transaction phases (from the information over the negotiation to the settlement phase); or the

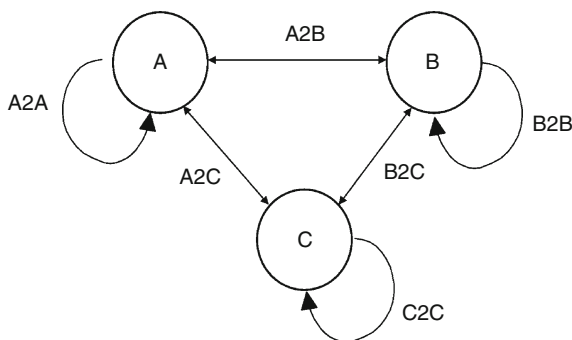
OECD with its definition referring to eCommerce as “business occurring over open, non-proprietary networks, including dedicated infrastructure; value generating activities within firms, suppliers and customers” [57]. This definition extends to the technology and infrastructure level and equates eCommerce with eBusiness.

On the other side, a more precise definition is given by the US Census bureau with eCommerce as a completed transaction (= agreement as transfer of ownership) over a computer mediated network, and with eBusiness being any process that a business organization conducts over a computer-mediated network (external and internal processes). This is similar to, for example [64], with eBusiness including eCommerce but also covering internal processes such as production, inventory management, product development, risk management, finance, knowledge management, and human resources. A similar view is provided by [72] with the statement that “it is important to note that e-business is far more than electronic commerce. E-business involves changing the way a traditional enterprise operates, the way its physical and electronic business processes are handled, and the way people work.” eCommerce is viewed as the online exchange of goods, services, and/or money, whereas – on an upper level – eBusiness automates all business processes and integrates them with eCommerce applications to create one seamless, digital enterprise serving customers and partners.

Independent from these different views one can classify the different eCommerce forms using the so-called eCommerce ABC (see [Fig. 19.1](#)).

Given these definitions, the objectives of applying eCommerce/eBusiness can be increased turnover by either of:

- Better customer relationships due to better communication
- More targeted and individualized marketing
- Better coordination between different marketing “instruments”
- New customer segments



■ Fig. 19.1

eCommerce ABC: (A) Administration, (B) Business, (C) Consumer/Citizen

or reduced costs by

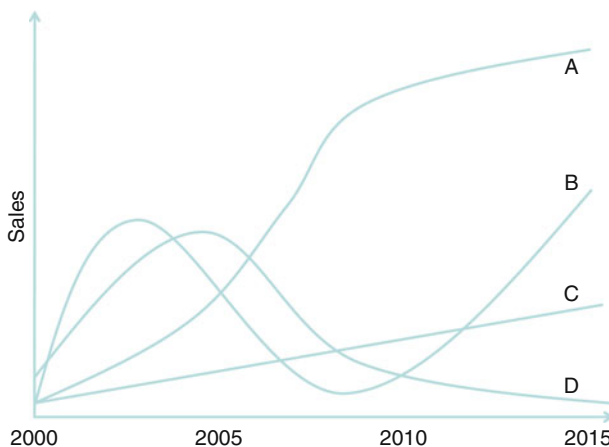
- Electronic distribution and the maintenance of digitized products and services (avoidance of media break)
- Customer self service (e.g., self check-in, electronic banking)
- Customer service 24 h a day, possibly worldwide

This chapter dwells on the two major domains of eBusiness/eCommerce, namely B2B and B2C. Regarding the former, a model-driven approach toward B2B IT solutions is introduced, covering semantic aspects dealing with business models, business process models, and business document models. Regarding the latter application domain, the basic concepts of the Semantic Web in the area of B2C eCommerce are introduced using an example from the eTourism domain.

19.1.2 A Short History and the Current Situation

When referring to the history of eCommerce it is interesting to compare qualitative growth forecasts with real numbers. [Figure 19.2](#) shows different potential trajectories for eCommerce sales [93]. Trajectory A follows an S-shaped growth curve (this was the assumption of many forecasts of the past), whereas trajectory B assumes rapid acceptance, followed by a sudden fall and then growth again. C shows slow and steady growth, whereas D grows rapidly and then declines.

However, numbers published by the US Census Bureau of the Department of Commerce (www.census.gov/mrts/www/current.html) support that, at least in the B2C field, there is something like a steady/linear increase, supporting trajectory C – as it was in fact really never assumed. The retail eCommerce sales for the fourth quarter of 2008 (not



■ Fig. 19.2

Qualitative predictions of eCommerce sales

adjusted for seasonal, holiday, and trading-day differences) were about US\$37 billion, or 3.8% of total sales. But these are overall statistics, where specific sectors perform much better. For example, in 2009, in the European travel and tourism industry approximately 26% of the total turnover will be carried out via the Web. And a survey of what Internet users worldwide research online reveals that the type of products/services they most searched for is holidays/destinations (61.9%), followed by consumer electronics and travel items such as flights/trains (www.newmediatrendwatch.com). The tourism industry is also a good example for the overall development in the B2C eCommerce area, where several system generations in few years (see [Fig. 19.3](#)) occur. This shows also the development toward customer-driven sites, or Web 2.0 applications.

In the travel and tourism industry, as in others such as the media and music industry, the online market was created by newcomers (either start-ups or companies from outside), with traditional players such as tour operators or airlines as careful observers. These stakeholders were constrained by their existing distribution channels as well as by integration problems in their legacy systems. This situation has changed dramatically into one of a hard competitive response of traditional players, using their market position in combination with existing distribution channels. This creates a very competitive situation – both with respect to technological as well as business innovations – including new and rather transient cooperation models between old and new players.

When looking at this B2B side of eCommerce, one observes an “informatization” of entire value chains from the initial supplier to the consumer, leading to flexible cooperation forms (value webs). Enterprise borders are blurred. These phenomena may eventually lead to so-called Smart Business Networks (see [Fig. 19.4](#)).

This also corresponds to the “move to the middle” hypothesis as an “intermediate” form between pure electronic markets and electronic hierarchies. Such network structures are organizational arrangements that use resources and/or governance structures from

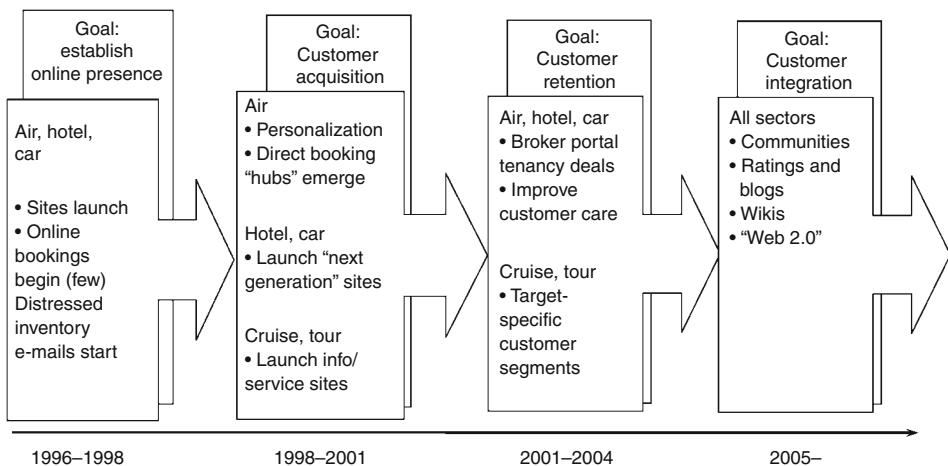
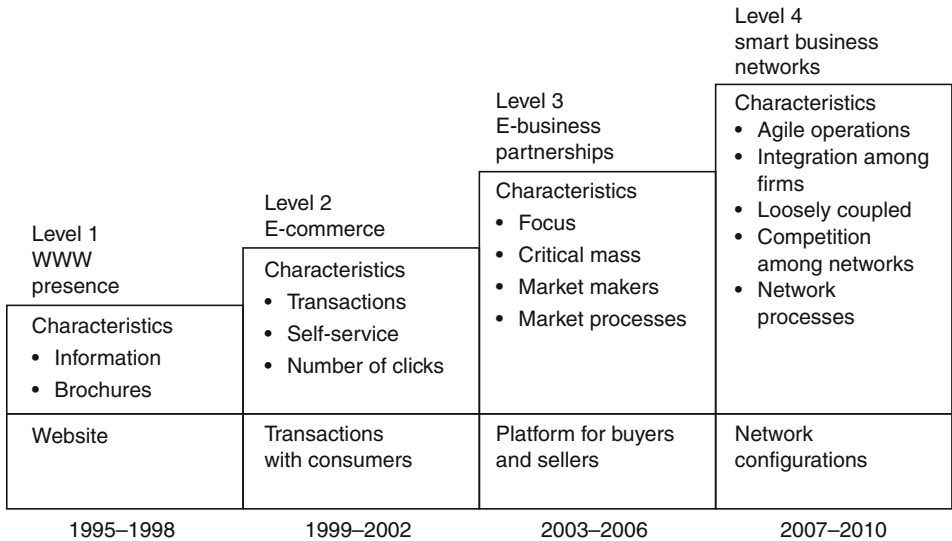


Fig. 19.3

Consumer view on eCommerce sites development (example tourism)



■ Fig. 19.4

Smart business networks as a phase of eCommerce development [91]

more than one organization. Long-lasting relationships protect against the loss of critical resources and enable the integration of activities. Such networks are a “middle way” between the loose coupling of markets and tight relationships of hierarchies.

On a qualitative level the current situation can be described as follows:

- B2C eCommerce shows an evolutionary development, where, for example, in Europe nearly half of the companies “accept orders online,” but this representing only a small portion of their total turnover; the Web is seen as an additional channel with the issue of multi-channel management.
- There are substantial differences between sectors (example tourism), where especially smaller companies lag behind; major issues are costs and know-how. This creates a form of a digital divide also within developed countries.
- In eCommerce there are small entry barriers easing the entrance of new/external “players.” Nearly all eCommerce markets were created by newcomers.
- Mobile applications have not seen the expected success; issues are “unclear” business models and not well-understood acceptance patterns.
- The at the beginning foreseen decreasing prices have only become partly true; counter issues are an overabundance of information, individualization strategies such as individual prices or recommendation applications, as well as clever branding campaigns by well-positioned online players. However, transaction fees are falling, which produces rather complex business models with product and service bundling [7].
- However, at the same time the Web and eCommerce leads to an immediate imitation of business models as well as features, which accelerates innovation. In tendency, this development, as well as eCommerce in general, favors users, see also Web 2.0.

- In general an Internet-based integration of processes may be observed, where eCommerce transforms industries. Currently, there is a strong trend toward the Web-based integration of services, provided by different providers. Services become standardized commodities which eases outsourcing and a deconstruction of value chains. This leads to the emergence of cooperative and/or virtual organizations. With this, the focus is not only on process reengineering, but also network engineering, based on a clear definition of the own value contribution. Complex market structures are emerging, with disintermediation and re-intermediation at the same time.

Thus, one can observe a permanent appearance of new services and at the same time a trend toward concentration with “the winners take it all” phenomenon (examples: search engine market or online travel agencies). With this, the evolution of the Web can be described as an ongoing interaction of order and disorder – on different levels:

- Structure with a tendency to concentration and the simultaneous entering of new players
- Services, where, for example, search engines can be seen as means to create order and, on the other side, individualized/recommendation systems or individual pricing
- Technology with periods of standardization, for example, the work of W3C or IETF, and then (or in parallel) technological breakthroughs

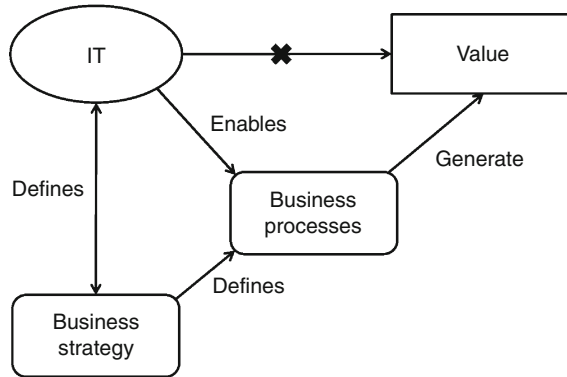
19.1.3 eCommerce Applications and Services

It is important to note that eCommerce applications or IT in general do not contribute directly to produce “value,” as already realized in the discussion regarding the well-known productivity paradox (see the economist Solow with his statement: I can see the computer age everywhere but in the productivity statistics). Following [98], one can understand the contribution of IT as indirect, via improvements of business processes and alignment of IT with strategy (see ► [Fig. 19.5](#)).

A similar approach is followed by Akkermans [3] in his logic of value, identifying three different dimensions which correspond to such an integrative view:

- “Techno logic”: looks at technology as well on its functionality and the capacity of companies to use it.
- “Business logic”: looks at business processes and the potential new roles of a company in (new) value chains.
- “Market logic”: looks at customers/suppliers interested, and related market structures.

Such an approach implies the combination of a business or a value view with an organizational as well as technical implementation. For such an integration a conceptual model crossing these various layers is needed. One possible approach is provided by service sciences [15]. It defines services as the starting point, putting the value exchange with the customer at the center. The underlying issue is to link strategy, business models, business processes, and implementation for the flexible design, implementation, and



■ Fig. 19.5

Relationship between IT, strategy, and business processes

adaptation of services. One should note that such an approach implies a transformation of meanings from services as they are understood in management science to Web Services as defined in Computer Science. In management science a service is defined as a business economic activity (intangible in nature), offered by one party to another to achieve a certain benefit [101], and “generated” by (internal) business processes. In information systems a service is seen as a simple or complex task executed within an organization on behalf of a customer [80].

The objective of this “service-oriented” view on eCommerce applications is to automate business processes and to improve user interactions. In this context eCommerce and eBusiness pose a set of challenges, on a technical, on an organizational, as well as on a market level [95]. Thus, in the remainder of this chapter not the entire eCommerce ABC is considered (see [Fig. 19.1](#)), but a focus is laid on B2B integration and on B2C, the latter with emphasis on the tourism domain as one of the main eCommerce application domains. [Sect. 19.2](#) deals with the application domain B2B, where the relationship of service-oriented architectures (SOA) and a model-driven approach toward B2B IT solutions is discussed. Three different types of concepts are discussed and linked with semantic approaches: business models ([Sect. 19.2.1](#)), business process models ([Sect. 19.2.2](#)), and business document models ([Sect. 19.2.3](#)). In [Sect. 19.3](#) Semantic Web concepts in the application area of B2C are presented. In particular the differences to the B2B world are discussed and a representative example, based on semantic technologies from the tourism domain, is presented.

19.2 Application Domain: Business-to-Business

Although the term eCommerce was coined during the dot-com boom, conducting electronic business between enterprises was not an invention of the Internet age, but has existed for decades. However, requirements of B2B electronic commerce have changed. In former days, when B2B electronic commerce was referred to as Electronic Data Interchange (EDI),

its focus was document-centric. This means, in order to avoid bilateral agreements on business documents, business partners agreed on business document standards. But, as history has shown, the results of these standardization efforts were overloaded and ambiguous document standards were created. This led to costly EDI systems and participation in electronic commerce was reserved to large companies that were able to afford such implementations.

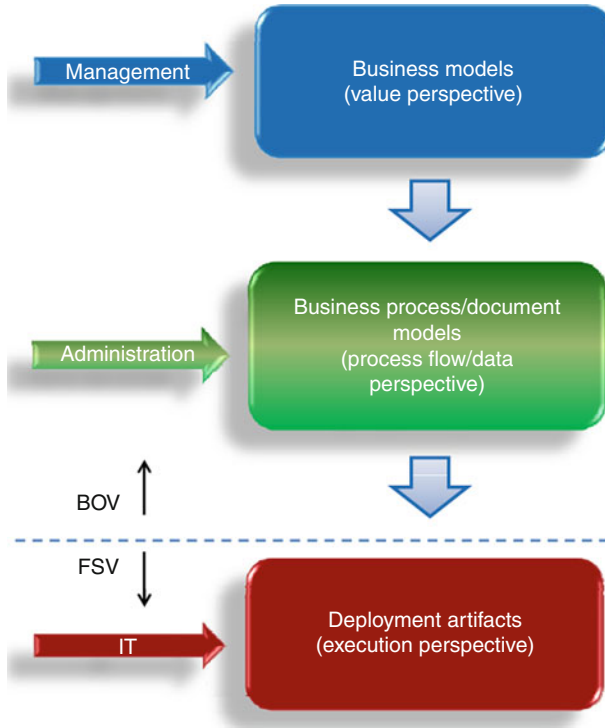
In today's business, companies must quickly adapt to faster changing business conditions. Business models must reflect these changes, business processes must be designed supporting the value exchanges, and IT applications must adjust to changing company goals. These requirements are often referred to as business/IT alignment. Management expects this to happen at low cost. Service-oriented architectures have the potential to provide a new level of flexibility in regard to the adaptation of the affected IT systems. Whereas in former days change requests to the IT resulted in a rigorous change of IT system implementations, nowadays service-oriented IT departments focus on the challenge of service alignment. Service alignment refers to the reconciliation between business partners to provide complementary services.

According to the OASIS SOA Reference Model [54], SOA stands for a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. This specification continues that in general, entities (people and organizations) create capabilities to solve the problems they face in the course of their business. The main drivers for a SOA are the management of the growth of large-scale enterprise systems, facilitating Internet-scale provision and use of services, and the reduction of costs in interorganizational cooperations.

It becomes evident that SOA is not limited to implementation issues addressed by Web Services – the current technology of choice to implement a SOA. A SOA-based approach to interorganizational cooperation must also address the business requirements in organizing and utilizing a distributed solution for a business partnership. This is in line with the Open-edi reference model, which became an ISO standard for interorganizational systems in 1997 [35]. Open-edi distinguishes between the business operational view (BOV) and the functional service view (FSV). The BOV addresses the business aspects such as business information, business conventions, agreements and rules among organizations. The FSV is related to information technology aspects, which are necessary to support the execution of a business collaboration. Accordingly, the BOV captures the business semantics which are implemented on the FSV.

The BOV layer has to capture the value propositions of each of the participating business partners [26] as well as the interactions between them. Consequently, separating the concerns in developing interorganizational systems results in three different perspectives shown in [Fig. 19.6](#). The management focuses on the value perspective described by business models. Business people have a process perspective described by business process models that operationalize the business models. The IT people focus on the execution perspective of the deployment artifacts implementing the business process models.

In the remainder of this section approaches to capture the semantics of the BOV, that is, on the business model layer and on the business process layer, are discussed. Since the



■ Fig. 19.6

Developing interorganizational B2B systems

business process layer has to address the business information being exchanged, business documents are introduced as well.

19.2.1 Business Models

As stated in the previous section, business modeling techniques are used to describe the economic drivers of B2B information systems. The growing popularity of the term business model is strongly interrelated with the Internet hype of the late 1990s [45]. At one stroke companies were able to increase sales by offering products and services 24/7 and simultaneously decrease transaction and procurement costs. For this reason the term business model quickly got popular and was used by a broad community, ranging from business people to scientists [44].

Paul Timmers defines a business model as “A business model defines an architecture for the product, services and information flows, including a description of the various business actors and their roles. Furthermore it describes the potential benefits for the various business actors and the source of revenues” [83]. Linder and Cantrell [44] share a similar point of view and define a business model as a company’s core logic in order to create value by

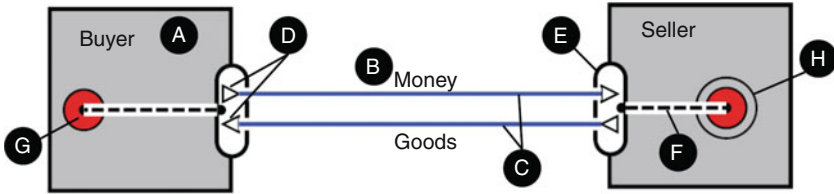
explaining how a company acts on the market and earns money. In addition they identified that a business model consists of distinct components which include and represent the essential business logic building blocks. These building blocks range from revenue models and value propositions to organizational structures and arrangements for trading relationships. Petrovic et al. [65] got one step further and formulated a hierarchical structure of distinct tiers of business logic spanning from business models over business process models down to information and communication systems. Therefore, a business model can be seen as a contextual link between business strategy, business administration, and ICT [59].

Business models have been defined and categorized in many different ways. Most authors rate their business models based on two dimensions. Functional integration and degree of innovation in case of Timmers [83], economic control and value integration by Tapscott [82], and the power of sellers and buyers by Pigneur [67]. Since the diversity of business model classifications shows the inadequacy of a unique classification scheme, Pigneur proposed another approach [68]. In contrast to the two-dimensional frameworks of Timmers, Tapscott, and Rappa, Pigneur suggests to use a multi-category approach. Thus, a single business model could be positioned in a Web of many classification schemas. They identified 12 principal dimensions for classifying business models: user role, interaction pattern, nature of the offerings, pricing system, the level of customization, economic control, level of security, level of value integration, value/cost offerings, scale of traffic, degree of innovation, and power of buyers and sellers.

In the following, three well-established business modeling techniques, e3-value [25], Resource-Event Agent (REA) [48], and the Business Model Ontology (BMO) [59] are introduced, all of them based on formal and semantic methods. Beside these well-established methodologies other approaches and frameworks (e.g., Business Engineering Model [49], The Edinburgh Enterprise Ontology [90], The Toronto Virtual Enterprise [20]) exist. However, due to space limitations not the whole range of available business modeling techniques is covered. For this reason the discussion is limited to the most important and well-accepted approaches. Furthermore, the business modeling techniques described in the following sections are suitable methodologies for the top-down approach in order to model interorganizational B2B systems starting from an economic point of view [17].

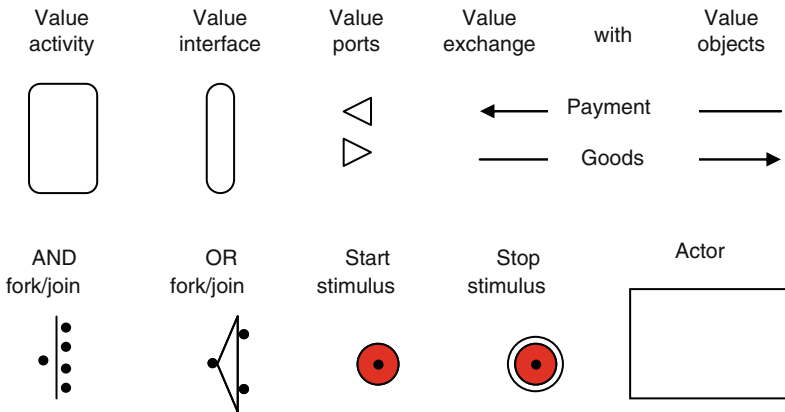
19.2.1.1 e3-value

The e3-value methodology has been developed to model a value Web consisting of actors who create, exchange, and consume things of economic value such as money, physical goods, services, or capabilities. It is an ontology-based methodology for modeling and designing business models for business networks, incorporating concepts from requirements engineering and conceptual modeling [25]. e3-value is based on the principle of economic reciprocity meaning a “give-and-take” approach between actors exchanging objects with an economic value – for example, if a seller delivers goods to a buyer, he gets money in return for the goods (see [▶ Fig. 19.7](#)).



■ Fig. 19.7

Simple example of an e3-value exchange



■ Fig. 19.8

The notation of e3-value

The graphical notation of e3-value comprises a small set of concepts and relations (see [▶ Fig. 19.8](#)) that have been introduced by Gordijn in [25]. Looking at the simple e3-value example in [▶ Fig. 19.7](#) actors are represented as rectangles (A). They are perceived by its environment as independent economic entities engaged in a value exchange. By exchanging value objects (B), they aim for either profitability (in case of an enterprise) or economic utility (in case of an end consumer). Value objects do not necessarily need to be a physical good. Sometimes they represent a service, right, or even a customer guarantee. A value object is always modeled in combination with a value transfer (C) and is represented as label. A value exchange is graphically modeled as a connection between actors.

Value objects are exchanges between actors using value ports (D). The concept of a value port is to signify whether the actor offers or requests a value object. Furthermore, it abstracts from the internal business processes, and a focus on how external actors and other components of the e3-value model can be “plugged-in.” Value ports are shown as small arrows pointing in the direction of the value exchange. A value interface (E) groups individual value ports. Each actor may have multiple value interfaces containing value ports for offering and requesting value objects. Value interfaces bundle the value objects an actor is willing to exchange in return for other value objects. The exchange of value objects via a value interface is atomic in order to denote reciprocity – that is, either all

exchanges occur as specified by the value interface or none at all. All concepts of e3-value discussed so far describe the inter-actor dependencies. In order to describe the intra-actor dependencies, scenarios are used to relate an actor's value interface. Such scenario techniques are described by so-called use-case maps (UCMs) and are used within the e3-value methodology in a simplified way. A scenario path (F) indicates via which value interfaces objects are exchanged. In order to keep track of a scenario path, the scenario path starts with a start stimulus (G) and ends with a stop stimulus (H). With these concepts a scenario path can pass through different actors being connected by a dotted line within an actor. AND forks as well as OR forks (and their corresponding joins) can be used to model two or more sub-paths.

It is important to stress that e3-value does not specify any order in time. This means that there is no order between the value exchanges within a value interface. Nor is there any order between the value exchanges of value interfaces connected by scenario paths. This is a significant difference between e3-value representing a business (value) modeling ontology and business process modeling approaches.

19.2.1.2 Resource Event Agents (REA)

The REA ontology was introduced by McCarthy [48] and extended by Geerts and McCarthy [22]. The concepts of REA reflect business accounting where the needs of managing businesses through a technique called double-entry bookkeeping was formerly the standard of use. REA represents double-entry with semantic models of economic exchanges and conversions. The acronym REA comes from the core concepts Resource, Event, and Agent. The intuition behind these core concepts is that every business transaction can be seen as an event where exactly two agents exchange resources. These basic REA concepts are illustrated in the cutout of the simplified REA meta model using a UML class diagram. [Figure 19.9](#) illustrates the simple Resource-Event-Agent structure at the meta-level from a conceptual point of view.

A business transaction or exchange is represented in REA by two paired economic events, noting that the two parties involved in a simple market transfer expect to receive something of value in return when they trade. For example, a seller, who delivers a product to a buyer, expects a requiring cash payment in return. In other words, in order to get

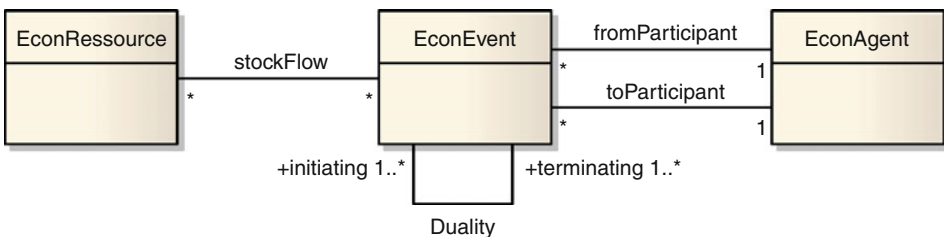
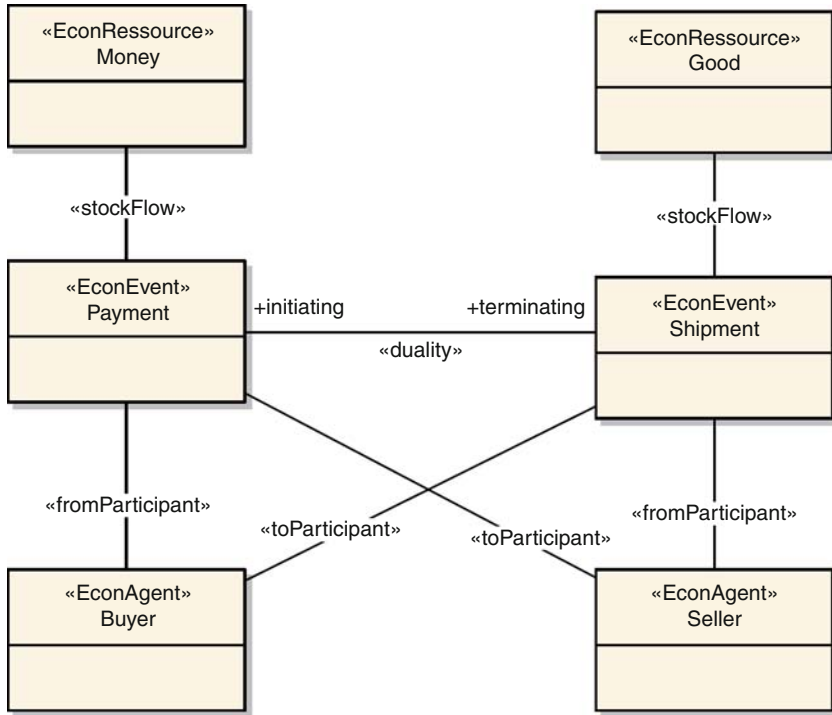


Fig. 19.9

The base of the conceptual REA meta model



■ Fig. 19.10

Simplified REA example of a buyer-seller scenario

a resource an agent has to give up a compensating resource. Figure 19.10 depicts a simple REA scenario covering the four fundamental questions of a business collaboration:

- *Who* is involved in the collaboration (Economic Agents – Buyer, Seller)?
- *What* is being exchanged in the collaboration (Economic Resources – Money, Good)?
- *When* (and under what trading conditions) do the components of the exchange occur (Economic Events – Payment, Shipment)?
- *Why* are the trading partners engaged in the collaboration (duality relationships between resource flows)?

As stated in the previous paragraphs, REA was initially designed for accounting and enterprise models. However, the REA concept has found its place in some standard specifications as well. The ISO Open-edi specification [35] uses REA as an ontological framework for specifying the concepts and relationships involved in business transactions. Furthermore, the REA ontology definitions are part of the work of UN/CEFACT (United Nations Center for Trade Facilitation and Electronic Business) which is an international eBusiness standardization body known for its work in the area of electronic data interchange (EDI) [87].


REA and e3-value share considerable overlaps, but also differences [6]. An example for such an overlap is the economic agent in REA and the actor in e3-value. In both

ontologies these concepts are used to describe participating business partners. Thus, a semantic mapping between e3-value and REA concepts becomes possible. In [74] the authors introduced mapping roles in order to translate e3-value models to REA models. Furthermore, they propose to use e3-value to depict the value network and REA to specify the economic drivers of the information system from a more IT driven perspective.

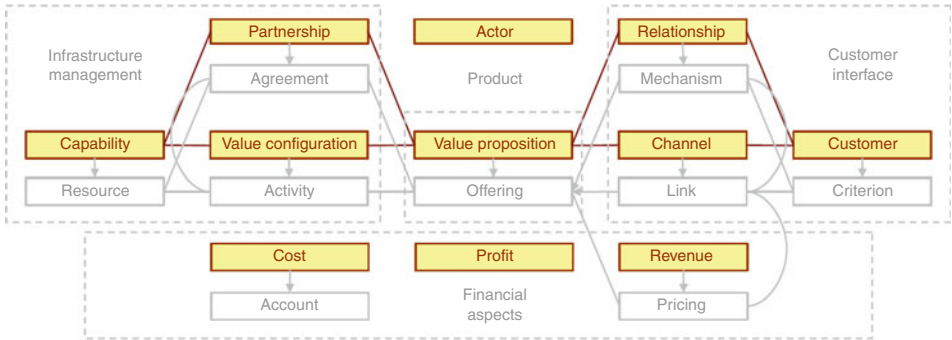
19.2.1.3 The Business Model Ontology (BMO)

The BMO has been introduced by Osterwalder and Pigneur [59]. They define a business model as a conceptual tool containing a set of objects, concepts, and their relationships with the objective to express the business logic of a specific enterprise. Therefore, it has to be considered which concepts and relationships allow a simplified description and representation of what value is provided to customers. Furthermore, it is important to foster how this is done and with which financial consequences. Thus, they found their ontology on nine concepts categorized into four main pillars. The categories are Product, Customer Interface, Infrastructure Management, and Financial Aspects.

- *Product* – The single building block of this pillar is the Value Proposition. A Value Proposition gives an overview about the product and services a firm offers, representing a substantial value to the customer.
- *Infrastructure Management* – The Infrastructure Management pillar consists of three building blocks: Value Configuration, Capability, and Partnership. The infrastructure and the network of partners that are necessary in order to create value and to maintain a good customer relationship.
- *Customer Interface* – This pillar is used to describe the target segment of customers. How the company wants to get in contact with these customers and which kind of relationship should be established between them. Thus, Infrastructure Management contains three distinct concepts: Target Customer, Distribution Channel, and Partnership.
- *Financial Aspects* – The financial aspects of a company, which are transversal and can be found throughout the three former components, such as cost and revenue structures.

Osterwalder splits the four pillars of the business model ontology into nine interrelated business model elements. While the four areas are a rough categorization, the nine elements are the core of the ontology.  *Figure 19.11* depicts the meta model of the Business Model Ontology and shows how the business model building blocks and pillars are interrelated with each other. A detailed description of the business model building blocks can be found in [4].

In contrast to e3-value and REA, BMO focuses on the position of a specific business partner in the eBusiness network and how he or she can make profit. Thus it depicts a business model from an internal point of view.



■ Fig. 19.11

The business model ontology

19.2.2 Business Process Models

Having reached an agreement on the value exchanges the second layer addresses the business processes to realize the value exchanges. First, the most popular notations for business process modeling, namely event-driven process chains (EPC), business process modeling notation (BPMN), and UML activity diagrams are introduced. Second, approaches that extend UML or EPC, respectively, in order to incorporate the business semantics are discussed.

19.2.2.1 Notation Approaches

Event-Driven Process Chains (EPC)

EPC is a modeling language to represent Business Processes graphically that was introduced in 1992. It is used for modeling, analyzing, and visualizing Business Processes in an enterprise. EPC is basically a directed graph connecting events and functions through control flows providing design abilities for parallelism. The small set of modeling elements provides an easy-to-understand model for business analysts as well as for management. IT specialists can use it as a basis for software development. ▶ [Figure 19.12](#) shows the basic elements of an EPC model.

Events. Events are passive elements and represent a changing state as a process proceeds. They can either be Start Events with external changes (trigger the start of a process), Internal Events with internal changes of states (changed by a process), or End Events with an outcome of a process that ends the chain and has external impact.

Functions. On the other hand, Functions represent activities or tasks of a business process and are therefore active elements. They are triggered by one or more events. Functions change the incoming state into the outgoing state. They are carried out by people or IT systems and may require resources.

In a process chain, Events and Functions have to alternate; therefore, an Event has to follow a Function and vice versa.

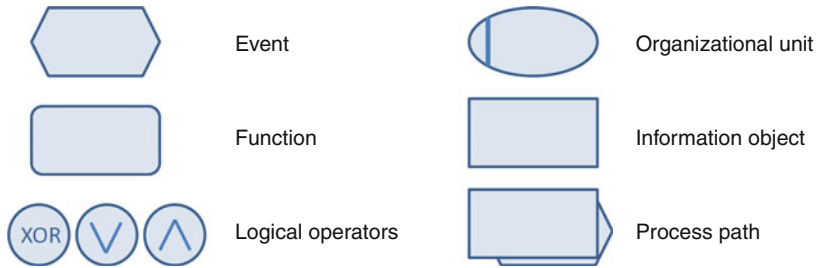


Fig. 19.12
EPC graphical elements

Logical Operators. Logical Operators can be applied in a process flow between Functions and Events. There are three basic types of Logical Operators: XOR, OR, and AND. Depending on whether they follow a Function or precede a Function these operators have different meanings. Following a Function (1) XOR will only follow one possible path, (2) OR will follow one or many paths, and (3) AND will split the path into multiple parallel paths. Preceding a Function (1) XOR will proceed by only exactly one incoming Event, (2) OR will proceed by any number of incoming Events, and (3) AND will only proceed if all incoming Events occur. If a path is split by any of the operators, it also has to be merged by the same operator later on in the process chain.

Organizational Unit. The Organizational Unit represents the person or organization, which is in charge of an associated Function.

Information Object. Information Objects supply input or output for Functions. These objects can, for example, represent a supplied or created document. An arrow indicates the flow of the information.

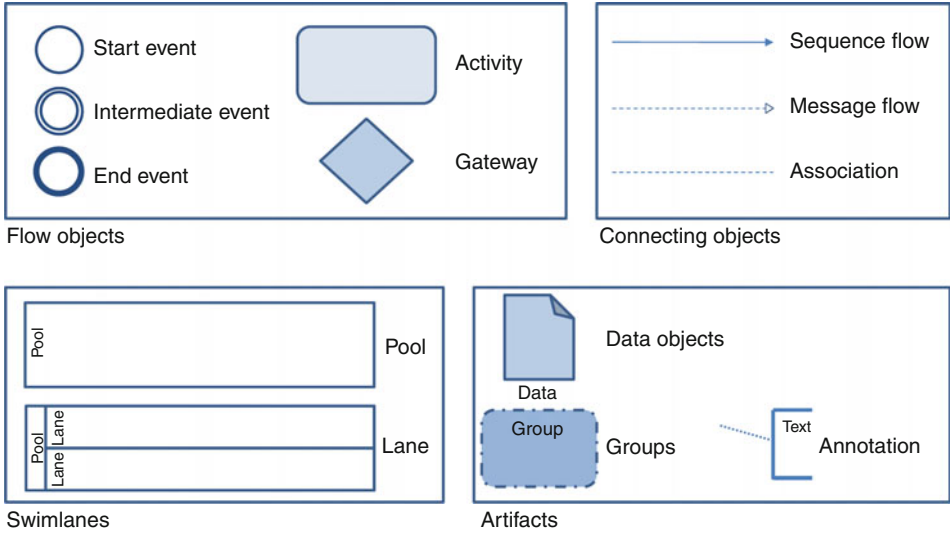
Process Path. Process Paths represent a complex activity and show the connection from or to other processes.

There are a couple of basic rules, according to [16], which an EPC model has to follow: (1) a model has to have one start and one end Event; (2) Events and Functions have to alternate; (3) each Function and Event has only one incoming and one outgoing path, except start and end Event; (4) decisions can only be made by Functions; (5) Functions making a decision are always followed by a Logical Operator; and (6) a single Logical Operator cannot have multiple incoming and outgoing paths.

According to [2], the following types of software tools use EPC: Business Process Reengineering tools, Enterprise Resource Planning systems, and Workflow Management systems. Actual examples for these tools are SAP R/3, ARIS, LiveModel/Analyst, and Visio.

Business Process Modeling Notation

Business Process Modeling Notation (BPMN) is a standard by the Object Management Group (OMG) to graphically represent Business Processes in a Business Process Diagram (BPD) and was first released in 2004 by the Business Process Management Initiative (BPMI). BPMN can either be used to describe internal business processes or collaborative



■ Fig. 19.13

BPMN graphical elements

B2B processes. The notation was developed in order to be understandable and used by the management, business analysts, and developers. Hence, there is no need for a transformation between a management and a developer view. As described in [62] the Business Process Execution Language (BPEL) can be generated out of BPMN. BPEL is based on XML and describes business processes connected through Web Services.

The graphical elements of BPMN described in [56] are arranged among four different categories (see ▶ Fig. 19.13): flow objects, connecting objects, swimlanes, and artifacts.

Flow Objects. An Event visualized by a circle represents something that is happening. The inside of the circle can contain an icon indicating the type of the Event such as a timer or message. Events can either throw or catch a message. The three different types of Events are Start, End, and Intermediate Event. Activities describe the work which needs to be carried out and are represented by a rectangle with rounded corners. They can either be a Task or a Sub-Process (indicated by a plus sign). A Gateway can split incoming Sequence Flows or it can merge them.

Connecting Objects. There are three different kinds of Connecting Objects: Sequence Flow, Message Flow, and Association. A Sequence Flow will define the sequence of the execution of activities. Message Flows show the message exchange between Actions or Events in different pools and can never be performed between two Actions or Events of the same pool. An Association is used to link text or Artifacts to Flow Objects.

Swimlanes. Swimlanes are used to organize and group Activities. There are two types of Swimlane objects: Pool and Lane. Pools may represent various participants like organizations. Other than Message Flows, a Sequence Flow is not allowed to link between different Pools. Lanes can divide a Pool into different subsections. They are used to organize Activities and may represent functions or roles.

Artifacts. Artifacts are used to provide additional information for the model. The three kinds of Artifacts are: Data Object, Group, and Annotation. Modelers are also allowed to create their own Artifacts if needed. Data Objects may represent data created or required by Activities and are connected to them by Associations. Groups are used to group together multiple activities for documentation and readability. Graphical elements in BPMN can be annotated with additional information. Therefore Annotations can be linked by an Association to an element.

This small set of modeling elements provides a fairly easy approach to create Business Process Models for current and proposed enterprise processes. According to OMG there are already 54 implementations of BPMN by companies such as IBM, SAP, FUJITSU, and SPARX SYSTEMS.

UML Activity Diagrams

An Activity Diagram is part of the Unified Modeling Language (UML) and describes what is happening in a workflow through a sequence of actions. It is mostly used for business process modeling, but can also be used for system modeling. It concentrates on a couple of graphical elements and supports parallelism and alternative paths through the workflow.

An Activity, which is represented by a rounded rectangle annotated by a name, includes a directed graph which consists of Activity Nodes and Activity Edges. Activity Nodes can either be Action Nodes, Control Nodes, or Object Nodes. Activity Edges are either Control Flows or Object Flows. To describe the activity flow tokens are used. They move along the edges from one node to the next node and show a possible flow. The main graphical elements of an Activity Diagram are shown in [Fig. 19.14](#).

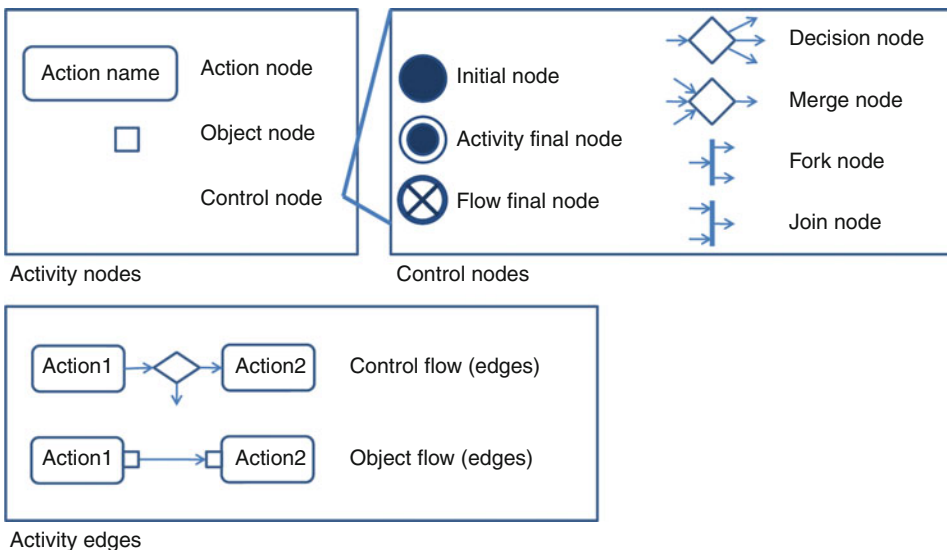


Fig. 19.14

UML activity diagram elements

Action Nodes receive and process input in order to create output for other nodes or call other activities. The name of the Action Node is displayed in the middle of the rounded rectangle. Object Nodes are used as formal input and output parameters. They are attached to either actions or activities. Control Nodes control the activity flow and are represented by a couple of different types: Initial Nodes are the start of a flow when the activity is invoked. Activity Final Nodes stop all the flows in an activity. A Flow Final Node just stops the arriving flow, but does not affect other flows in the activity. A Decision Node has one incoming and multiple outgoing flows, but only one of them will be chosen if the token arrives. Merging these multiple flows is carried out by the Merge Node. A Fork Node splits the incoming flow into multiple concurrent outgoing flows. Concurrent flows can be again joined into one single flow by a Join Node. There are two types of Activity Nodes, which can also be annotated by a name close to the arrow head. One of them is a Control Flow, which will start one activity after the previous one is finished. Data and objects are not allowed to be passed.

Contrarily, an Object Flow can pass data or objects from one Object Node to another.

The simplicity of UML Activity Diagrams makes it an easy and intuitive model to create and understand business processes and it is widely supported by various tools.

19.2.2.2 Semantic Approaches

UN/CEFACT's Modeling Methodology (UMM)

In general, one may use any of the above-mentioned notations to model B2B processes. However, B2B processes have certain well-defined characteristics that distinguish them from intra-organizational processes. Thus, it is important to consider the special B2B semantics in a corresponding business process modeling approach. In other words, a rather general business process language must be further constrained to the B2B semantics. The UN/CEFACT Modeling Language – which is further detailed here – does this by putting UML into the corset of B2B semantics.

In order to enable two business partners to engage in an automated business transaction, an agreement on a common process choreography and on a common business document data definition is needed. The United Nation's Centre for Trade Facilitation and Electronic Business (UN/CEFACT), known for its standardization work in the field of UN/EDIFACT and ebXML [55], took up the endeavor and started research for a methodology for process choreographies and business document definitions. This ongoing work resulted in UN/CEFACT's Modeling Methodology (UMM). UMM enables one to capture business knowledge independent of the underlying implementation technology, like Web Services or ebXML. UMM is used to model an interorganizational business process concentrating on the flow of interactions between collaborating business partners. However, it does not address their private processes. In general, the execution of

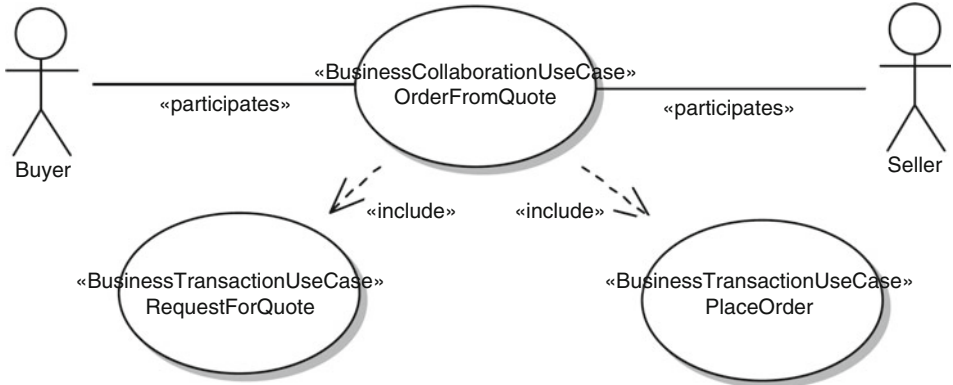
an interorganizational business process depends on commitments established between the participating partners. UMM is used to model these procedural and data exchange commitments that must be agreed upon between the business partners at design time of the interorganizational business process, that is, before executing the process. Inasmuch, the UMM model becomes a kind of “contract” that guides the business partnership. Since most commitments are made between two partners only, a UMM model – such as most contracts – is agreed upon between two parties.

Also similar to a contract, a UMM model describes the commitments on the information flow from a neutral perspective. It follows that UMM describes a bilateral and global choreography. UMM is defined as a UML profile [85], that is, a set of stereotypes, tagged values, and constraints – in order to customize the UML meta model to the special purpose of modeling global B2B choreographies.

The UMM follows a well-defined development process that produces a set of well-defined artifacts. The development process runs through three major phases, which correspond to the three top-level packages of UMM: the business requirements view, the business choreography view, and the business information view. In this section requirements are not elicited using the business requirements view of UMM, but are built upon the requirements already gathered by the REA and e3-value models, which were introduced in the last section. The reader interested in all details is referred to the UMM paper in [34] as well as to the specification [87]. In this section, artifacts of two sub-views of the business choreography view, the business transaction view and the business collaboration view, are introduced. A business collaboration view comprises the artifacts of a business collaboration, which spans over multiple business transactions. A business transaction view covers the artifacts of business transactions, which is a business information exchange between two partners including an optional response. The information exchanged within a business transaction is modeled in the business information view, which is outlined in detail in [Sect. 19.2.3](#).

In order to exemplify the UMM artifacts a simple order from a quote example is used for demonstration purposes. In this example a buyer requests a quote from a seller. Once he receives it, he is able to order products, which is confirmed by an order response. The order from quote business collaboration between the buyer and the seller consists of two business collaborations: request for quote and place order. [Figure 19.15](#) depicts the corresponding use cases. A UMM business transaction view is characterized as follows:

- A business transaction view includes exactly one business transaction use case and the two authorized roles participating in this use case.
- A business transaction use case is the parent of exactly one business transaction that is modeled by an activity diagram.
- A business transaction is built by two partitions, that is, one for each participating authorized role.
- Each partition includes exactly one business action, either a requesting business action or a responding business action.



■ Fig. 19.15

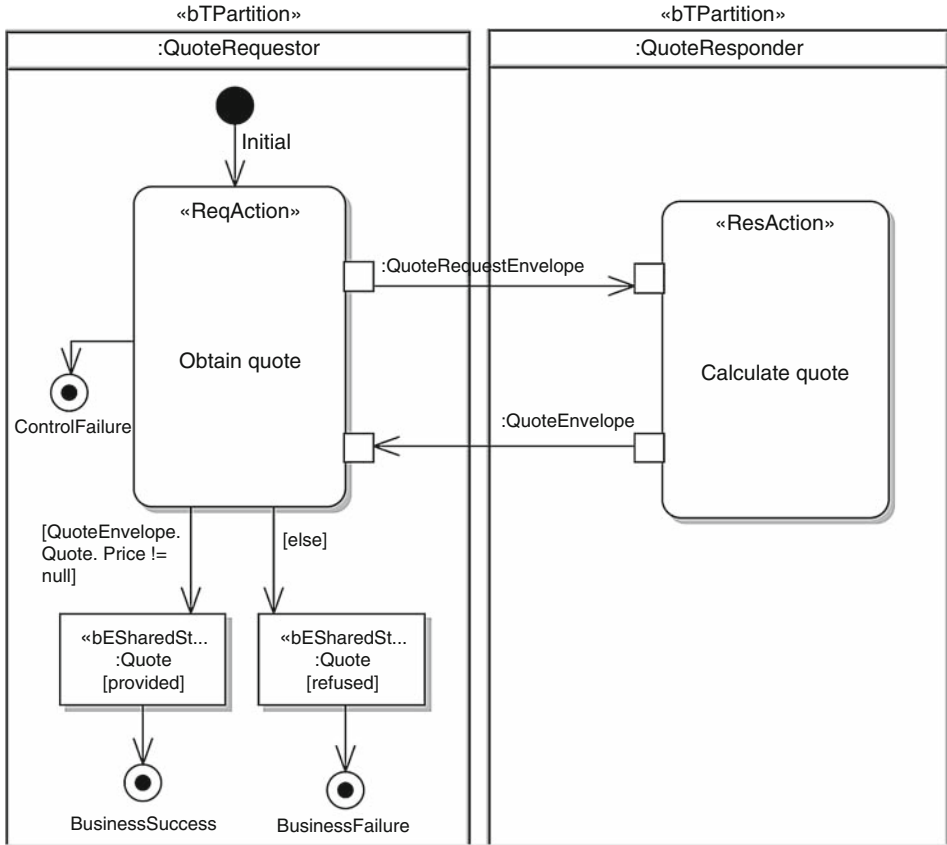
Business collaboration/transaction use cases

- Exactly one requesting business information envelope is exchanged from the requesting business action to the responding business action.
- Zero, one, or one out of more alternative responding business information envelopes is returned from the responding business action to the requesting business action.
- The requesting business action leads to one or more alternative business entity states depending on the received results.

This example involves two business transaction views. The first one, request for quote, comprises the business transaction use-case request for quote – depicted on the lower left of [Fig. 19.15](#) – which is the parent of the business transaction in [Fig. 19.16](#). This business transaction follows the pattern described in the bullet list above. A quote requestor performs obtain quote which sends a quote request envelope to the calculate quote action of the quote responder. A quote envelope is later returned to the obtain quote action. Depending on the result – whether a price is given in the quote envelope or not – the business transaction leads to one of the two states of the business entity quote: provided or refused. The second business transaction view covers the business transaction use-case place order (see [Fig. 19.15](#)) and its corresponding business transaction. This business transaction – which is not depicted in here – follows again the same basic pattern.

A UMM business collaboration view is characterized as follows:

- A business collaboration view includes exactly one business collaboration use case and the authorized roles participating in this use case.
- A business collaboration use case includes other business collaboration use cases and/or business transactions cases, each defined in its own business transaction view.
- A business collaboration use case is the parent of exactly one business collaboration protocol that is modeled by an activity diagram.

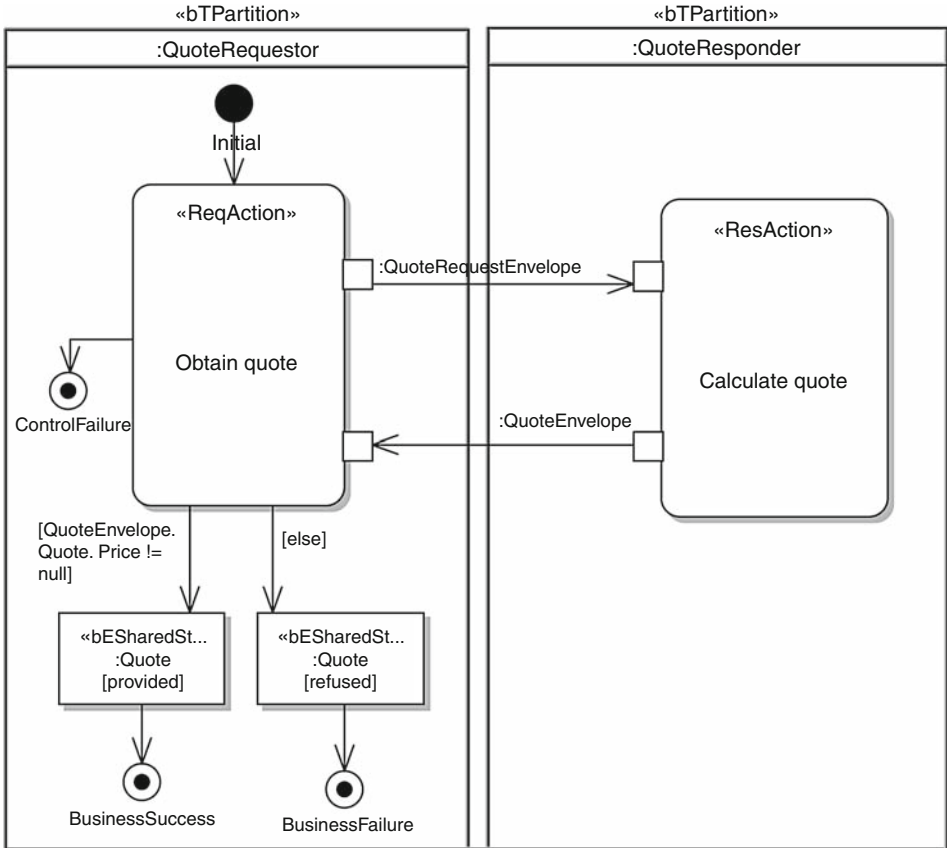


■ Fig. 19.16

Business transaction: request for quote

- A business collaboration protocol is built by business transaction actions and business collaboration actions. The former calls a business transaction. The latter calls another business collaboration protocol.
- A business collaboration protocol includes a partition for each participating authorized role.
- Information flows – the init-flow and the optional re-flow – connect each business transaction action with the partitions in order to denote who is playing which role in the underlying business transaction.

In this example, the first business collaboration view includes the business collaboration use-case order from the quote depicted on top of [Fig. 19.15](#). This use case is the parent of the business collaboration protocol of [Fig. 19.17](#). This business collaboration protocol includes two business transaction actions. The first one calls the business transaction request for quote. If this business transaction sets the business entity quote



■ Fig. 19.17

Business collaboration protocol: order from quote

to state “refused”, the business collaboration protocol ends with a failure. If it sets it to “provided”, the business collaboration protocol continues with a call of the business transaction place order. Depending on whether the order is set to accepted or rejected, the business collaboration protocol ends with a success or a failure. The init-flow starts for both business transaction actions from the buyer. Thereby, it is denoted that the buyer is the initiator of the underlying business transactions, which is the quote requestor in the first transaction and the purchaser in the second transaction. The init-flow leading from the business transaction actions to the partition of the seller denotes that he or she is the respondent in the respective underlying business transactions. The re-flows indicate that the underlying business transactions comprise a bidirectional message exchange. If an underlying business transaction has an unidirectional message flow (e.g., in case of a notification), the re-flows are omitted.

BSopt

As a further example the BSopt (<http://www.bsopt.at>) project is presented, which uses UMM as an essential technique to describe the semantics of business processes. BSopt (Business Semantics on top of process technology) is funded under the Semantic Systems Program of the Austrian Research Promotion Agency. It develops a methodology and a tool set for a top-down approach for SOA where the business requirements drive the underlying IT infrastructure implemented by Web Services. In other words, BSopt defines a methodology that considers business models, business process models, and deployment artifacts for a SOA. Business models are expressed by the means of e3-value and REA. On the business process layer BSopt distinguishes approaches to model the global choreography of an interorganizational system and approaches to model the internal processes that provide an interface to the choreography. For global choreographies, BSopt uses UMM and integrates concepts of BPMN. For modeling internal processes UMM concepts are taken and extended by UML profiles that are based on UML activity diagrams. In order to make business process descriptions machine-interpretable, BSopt generates deployment artifacts capturing the process specifications. BSopt uses BPEL (Business Process Execution Language) as a definition language to generate abstract processes. The top-down approach of BSopt is depicted in [Fig. 19.18](#).

BSopt integrates the approaches on each of the three layers and specifies a meta model. A model created by the BSopt methodology must be in accordance to this underlying BSopt meta model. The meta model integrates all the semantics concepts on each layer

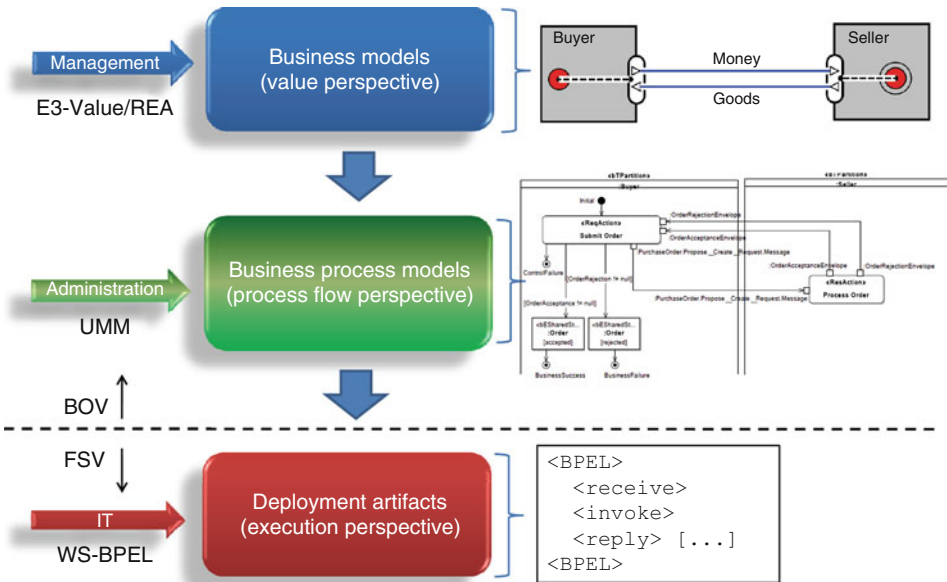


Fig. 19.18
The three-layer architecture of BSopt

and their interdependencies. Since business models and business process models are often specified by different notations the conceptual BSopt meta model is defined by a uniform language. In order to cope with the different requirements on each level BSopt uses a flexible knowledge representation language. Thus, the conceptual meta model of BSopt is expressed in OWL.

SUPER

As a second example of demonstrating semantics the SUPER (<http://www.ipsuper.org>) project is introduced. The major objective of SUPER (Semantics Utilized for Process Management within and between Enterprises) is to raise business process management from the IT level to the business level. This objective can only be achieved, if business process management is accessible to business experts and business analysts without requiring detailed technical expertise. SUPER uses Semantic Web and particularly Semantic Web Services in order to enable users to perform complex tasks without requiring an understanding of the underlying technology. SUPER aims at providing a framework that is context-aware based on Semantic Web Service technology, and which acquires, organizes, shares, and uses the knowledge embedded in business processes and IT systems. Business experts and analysts can access this knowledge in an understandable format through a process modeling tool, which is tailored for the use of SUPER concepts. The tool enables them to easily analyze, change, and create business processes, leading to a higher degree of agility in companies [12].

SUPER achieves this objective by adding semantic annotations to business process modeling artifacts (like process activities, services, and execution artifacts), making these artifacts accessible for advanced querying and reasoning [46]. Using these querying and reasoning approaches, the tools developed in SUPER support users during business process modeling through techniques such as Semantic Business Process Discovery [47], Semantic Business Process Composition [94], and Semantic Business Process Mediation [52]. Semantic Business Process Discovery supports the business expert during the business process modeling phase by simplifying the reuse of existing artifacts. In order to find these artifacts, SUPER provides a formal framework for the description of business process models. The business expert can expose expressive queries to the business process repository to search for existing process components. Semantic Business Process Composition provides a mechanism that enables business experts to operationalize their business processes directly, by automatically deriving an executable process from a conceptual business process model. Semantic Business Process Mediation facilitates Semantic Business Process Composition by enabling the seamless integration of processes originating from various stakeholders in a collaborative business process. The use of such technology facilitates the task of modeling business processes in two ways: first, it improves the quality of the models through the reuse of established and optimized process components; and second, it reduces the process modeling time by avoiding reinventing the wheel. Furthermore, Business Process Mediation deals with heterogeneity in the behavioral interfaces and message formats of processes. This means that a Mediation Service converts the different message formats that are being exchanged between two or

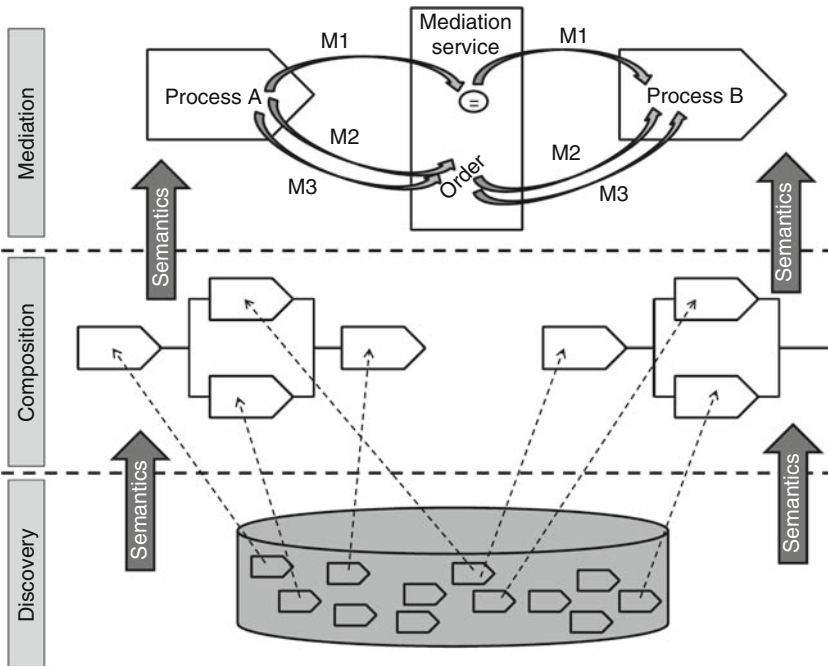


Fig. 19.19
The essentials of the SUPER project

more processes into a common understandable format. [Figure 19.19](#) depicts the three essential parts of the SUPER framework.

Comparing BSopt and SUPER

There are some overlaps as well as differences between the BSopt and the SUPER approach. As stated in the previous paragraph SUPER annotates business processes by the means of semantic concepts. The BSopt approach describes the semantics of business process models by means of business modeling techniques – for example, e3-value or REA. The same concept is applied to the service implementation layer. Whereas in SUPER, service compositions are semantically annotated, BSopt describes the IT layer by the business process modeling concepts of UN/CEFACT's Modeling Methodology (UMM).

In contrast to SUPER, the BSopt approach considers the economic drivers of a business process and evaluates the economic sustainability of the IT system to be designed. This is achieved by the use of the business modeling technique e3-value. The methodology does not only help to draw a first sketch of the economic values exchanged in an eBusiness network, it also offers the possibility to calculate so-called profitability sheets by quantifying the net value flow for each actor in the value Web. In order to develop a business model for an information system, BSopt proposes to start with e3-value and then map down to REA. In [74] conceptual rules for mapping an e3-value model to an REA model are introduced. Whereas e3-value concentrates more on the

profitability of the IT system, an REA business model focuses on issues that may be relevant for the implementation and alignment of an IT system.

19.2.3 Business Documents

The last section has outlined the importance of an agreement over a common business process model. While a process choreography defined by a process model describes the exchange order of business documents in detail, little is said about the harmonization of business information that is being exchanged. Thereby two different domains of information are distinguished: business document information and product data–related information.

In particular in the field of interorganizational business processes, business documents such as invoices, purchase orders, etc., are vividly exchanged between different organizations. Seamless integration between the involved IT systems is only possible if all business partners have a common understanding of the exchanged business document information. Additionally, a common definition of product data–related information is of importance. Goods such as computer hardware, livestock, vegetables, etc., are transferred from the manufacturer via wholesalers, retailers, etc., to the end consumer. Most of the involved business partners in such a supply chain use different representation and encoding mechanisms to represent the traded good in their IT systems. If a common representation format for the exchanged goods could be provided, a smoother integration in the different IT systems would be possible.

In this section the current state of the art in the definition of a common business document ontology is examined. First, a historical overview of business document standards is given and the core component approach for the definition of a common business document ontology is introduced. Finally, the concept of product data catalogs is introduced and the differences compared to business document standards are examined.

19.2.3.1 Business Document Standards

In the early days of computing, document standardization focused on the common definition of file formats, which were vendor specific and mostly used a binary representation format. With the proliferation of distributed systems new interchange formats were required, which are valid across company boundaries. The industry as well as vendors soon recognized the need for a common interchange format definition and started to work on business document standards.

➤ *Figure 19.20* gives an overview of the most important standard definitions of the last 40 years. The black dots are standards, which are delimiter based, that is, regular ANSI characters are used to separate different segments and data elements. One of the best known delimiter-based approaches for the standardization of exchanged data is UN/EDIFACT, maintained by the United Nations Center for Trade Facilitation and

Electronic Business (UN/CEFACT). The UN/EDIFACT standard provides a set of syntax rules, used to structure business document data. The document format uses designated symbols and letter codes as delimiters between the different data fields. Another delimiter-based approach, merely focusing on the North American market, is ANSI X12. Delimiter-based solutions were mostly developed in the 1980s and early 1990s of the last century.

Based on the General Markup Language (GML), which was developed by IBM around the year 1970, the Standard General Markup Language (SGML) was developed. SGML served as the basis for the development of the eXtensible Markup Language (XML), which itself is a subset of SGML. Other standards derived from SGML include the Hypertext Markup Language (HTML) and the eXtensible Hypertext Markup Language (XHTML).

Since the introduction of XML in 1996, its popularity has constantly increased due to its versatility, flexibility, and easy applicability. One of the major application fields of XML was the definition of common interchange formats. Using the markup concept users could easily define their own dedicated data structures. Technologies such as Document Type Definitions (DTD) and XML Schema guaranteed the well-formedness and validity of XML documents. Standardization organizations recognized the potential of XML and started to develop their standard definitions using the new markup language. Standards based on markup languages are denoted by white dots in [▶ Fig. 19.20](#). In the mid-1990s a clear transition from delimiter-based standards to markup-based standards is observable. Although delimiter-based standards are still in use today, XML-based solutions are the current state-of-the-art solution for business document standard definitions. An overview of different XML-based standards for describing data and business documents is given by Li [40]. An additional boost for XML-based solutions has been brought by the introduction of Web Services and their related technologies such as Web Service Definition Language (WSDL), Simple Object Access Protocol (SOAP), and Universal Description, Discovery, and Integration (UDDI). In particular, in the context of Web Services the clear and precise definition of a business document is of importance. Usually interfaces defined by WSDL import the appropriate XML schema defining the type of business document the interface accepts. The relationship between Web service standards and the Semantic Web is covered in [▶ Semantic Web Services](#).

Core Components

Given the popularity of XML as the representation format of choice for data, several initiatives have been started in order to standardize exchanged data using XML. However, the transition from a delimiter-based approach such as UN/EDIFACT to an XML-based solution did not solve the interoperability problems between business documents. In the following, shortcomings in regard to business document standardization are addressed and it is shown how Core Components tackle these issues:

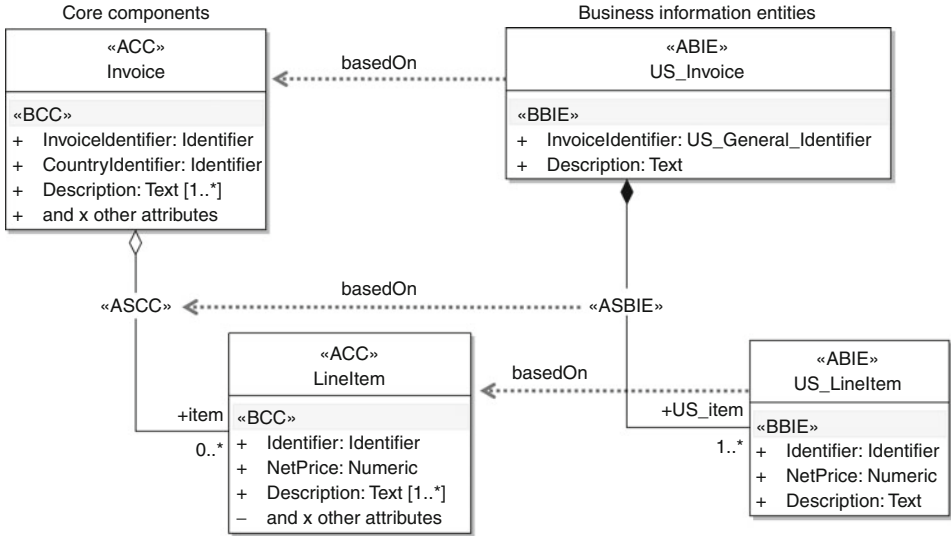
1. *Standard incompatibilities.* Due to the multiple initiatives that have been started, several XML-based standards are now directly competing with each other. Furthermore, the multitude of standards results in large incompatibilities between the

different definitions – a fact that initially was to be solved with the introduction of business document standards.

2. *All-in-one approach.* Furthermore, a lot of standards aim at the integration of every possible element into a standardized business document, resulting in a significant document overhead. For example, a cross-industry invoice which should be applicable in any industry context has to include every possible element that any of the different industries might need. Whereas for instance `number of nights per person` is critical in a tourism context, this attribute is rather unlikely to be needed in an oil industry context. However, in order to be cross-industry compatible every possible element has to be included in the standardized invoice. A partial solution is given by so-called message implementation guidelines, which cut down a standard to a set of agreed elements that are used between two business partners. However, this results in a multitude of message implementation guidelines undermining the concept of a holistic business document standardization.
3. *Transfer syntax specific definition.* Standards such as UN/EDIFACT or XML-based solutions for business documents are tightly bound to the implementation syntax. Often the document semantics are defined on the logical level (e.g., XML schema) instead of being defined on a higher, conceptual level. Changes in the transfer syntax, therefore, result in reengineering tasks for the standard, making it inflexible to future adaptations.
4. *Conceptual document description.* Business document standards mostly lack a unique conceptual description model, but merely focus on implementation details. Whereas this approach is sufficient for implementation, it is hard to communicate a logical level model such as XML schema between different modelers. A standardized representation mechanism for the communication of business document concepts is needed.
5. *Missing ontological representation.* Most of the business document standards do not provide an RDF or OWL representation, and thus document processing using Semantic Web tools is difficult.

In order to build a global business document reference ontology UN/CEFACT started the development of the so-called Core Components Technical Specification [89]. The idea is to develop a common ontological base of reusable building blocks for business documents. Using these building blocks, a shared library is built from which modelers can retrieve artifacts in order to assemble a business document.

The development of the core components standard started in the late 1990s as part of the ebXML [55] initiative. The main goal of ebXML was to provide a framework allowing potential business partners to engage in B2B processes in an interoperable, secure, and consistent manner. One part of the ebXML technology stack were so-called core components, used to uniquely define the exchanged data between two enterprises. UN/CEFACT's Technologies and Methodologies group continued the development of core components and today the standard is known as the Core Components Technical Specification (CCTS). The most recent version of the standard is 2.01 [89] with the development of version 3.0 [86] currently ongoing.



■ Fig. 19.21

Simple core component example

► Figure 19.21 shows a simple core component example using the UML syntax as introduced by the UML Profile for Core Components (UPCC) [88]. The core component approach differentiates between two elementary concepts: context-neutral and context-specific artifacts. As shown on the left-hand side of ► Fig. 19.21, core components are defined independent of any business context on a context-neutral level.

The context-neutral core component basis is used to derive context-specific artifacts for certain businesses as shown on the right-hand side of ► Fig. 19.21. If core components are used in a certain business context they become business information entities. Since the derivation of business information entities from core components is only possible by restriction, a business information entity will only contain attributes and associations which have already been defined in the underlying core component. Thus, all business information entities share the same semantic basis.

An introduction to the Core Component business document ontology is provided in [42] and the OWL representation for Core Components is introduced in [43]. Apart from the domain of Web Services, standardized definitions also play an important role in the field of product catalogs. In the following, state of the art in this domain is introduced.

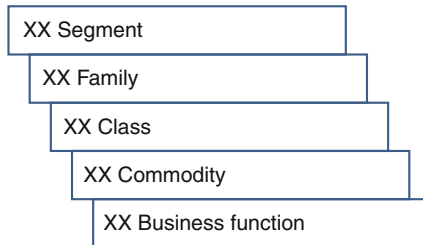
19.2.3.2 Product Data Catalogs

Product data catalog ontologies have gained considerable attention in the semantic community in the last few years. In short, the goal of product data ontologies is the reuse of data across enterprise boundaries. Using a unified view on commerce-related data on the Web allows for complex semantic search queries, which go beyond traditional

approaches. A quantitative analysis of different product categorization standards has been conducted by Hepp et al. [32]. In their analysis the authors examine different categorization standards namely, eCl@ss, United Nations Standard Product and Services Code (UN/SPSC), Electronic Open Technical Directory (eOTD), and the RosettaNet Technical Directory.

One of the most promising product data catalogs is the UN/SPSC standard. UN/SPSC is based on the United Nation's Common Coding System (UNCCS) and the Dun & Bradstreet's Standard Product and Services Classification Code (SPSC). In a nutshell, UN/SPSC provides a hierarchical classification with five levels. ▶ [Figure 19.22](#) gives an overview of the UN/SPSC structure.

Each level of the hierarchy has its own unique number. A segment denotes the logical aggregation of families for analytical purposes. Families represent a commonly recognized group of interrelated commodity categories. A class is a group of commodities sharing common characteristics. Commodities are substitutable products or services. Business functions are functions, which are performed by an organization in support of the commodity. ▶ [Table 19.1](#) shows a UN/SPSC example with explanations.



■ [Fig. 19.22](#)
UNSPSC overview

■ [Table 19.1](#)
UNSPSC example

Hierarchy	Category	Number and name
Segment	43	Information technology broadcasting and telecommunications Communication Devices and Accessories
Family	20	Components for information technology for broadcasting or telecommunications Computer Equipment and Accessories
Class	15	Computers Computer accessories
Commodity	01	Computer switch boxes Docking stations
Business function	14	Retail

Since the data in [Table 19.1](#) are defined in a unified manner using an open global standard, it can easily cross enterprise boundaries on an international level. Thus, effective and automated procurement and sales coordination between enterprises is possible.

Another promising approach toward the definition of a common product ontology is the Good Relations Ontology [31]. The ontology aims at the definition of a universal and free web vocabulary for product-related data. The vision is to provide one single schema for a consolidated view on electronic commerce data. Thereby, the single view on the commerce-related data provides a set of benefits for the involved business partners. Web shops may gain better visibility in latest generation search engines, since their ontology annotated data allow for more complex and precise search queries. Not only are goods described in more detail, but also terms and conditions of items and services offered on the Web may be described. As an example, a particular website describes an offer to sell MP3 players of a certain manufacturer at a certain price. In contrast, a car dealer may not only offer cars of a certain brand, but also maintenance services for certain brands and types of cars. Manufacturers benefit from the fact that product feature data may be exchanged and reused together with retailers with a minimal overhead.

Currently a successful implementation of the Good Relations Ontology has been provided for Yahoo! SearchMonkey. Using the Yahoo! SearchMonkey site owners and developers may use structured data to make search results by Yahoo! more meaningful.

19.3 Application Domain: Business-to-Consumer

In the last years, household access to broadband Internet has been diffusing rapidly. In 2007, more than two-thirds of all households had access to broadband Internet in countries such as Denmark, Finland, Iceland, and the Netherlands (OECD). The increase of broadband access has changed the role of the Internet. Formerly being regarded as a pure information search channel, it has turned to a medium that enables communication between people (such as e-mail, instant messaging, or community portals), buying physical goods (books or clothing), digital goods (music, games, or software), as well as purchasing tickets for events or travel-related items (flights or hotels); 875 million people, that is, 85% of the world's online population, have bought a product online, which represents an increase of 40% over the last 2 years. According to the Eurobarometer results 2008, 33% of all consumers in the EU27 have purchased goods or services via the Internet in 2008. eCommerce is therefore the most popular distance selling channel in the EU. Online commerce in Europe is expected to rise from \$197 billion in 2007 to \$406 billion by 2011. In fact, 51% of EU27 retailers sell via the Internet (Eurobarometer 224).

One of the industries that has gained enormously from the use of the Internet is the tourism sector. Internet technology has created an online travel market where tourism businesses are able to sell their travel products and communicate with their customers through electronic media. The wealth of information available online has thus empowered consumers exploiting the Internet to research travel-related information and book part of

their trip online, thereby undercutting the business of the traditional travel agencies. C. Marcussen (Centre for Regional and Tourism Research) states that online travel sales increased by 17% from 2007 to 2008 and reached 58.4 billion in the European market in 2008, i.e., 22.5% of the market. With the rise of Web 2.0, the amount of travel community platforms and travel sites exploiting user generated content (UGC) has exploded. Next to social networking platforms, mobile applications and services will become an increasing important element in the near future. According to an EyeForTravel Research, 73% of travel companies are convinced that mobile technology will change the way they will communicate with their customers.

19.3.1 The Tourism Industry and Its Characteristics

The tourism industry is not only an adopter of Internet technology but also a driving force in the eCommerce sector. However, being a worldwide industry, it is exposed to the current global economic crisis as well. Consequently, the negative trend in international tourism that emerged in 2008 has intensified in 2009 and resulted in a drop of international tourist arrivals (UNWTO World Tourism Barometer). But looking beyond the crisis, the tourism industry will continue growing in importance as one of the world's highest priority industries. Indeed, the travel and tourism economy accounts for more than 9% of the global GDP (World Travel and Tourism Council).

It represents a cross-sectoral (umbrella) industry, containing a variety of economic sectors such as culture, gastronomy, accommodation, or transportation. These different industrial components form a networked industry, where the production and distribution of tourism goods is based on cooperation. The tourism product is a complex good, as it is often provided to the consumer in the form of a travel package that comprises basic products such as flights, cars, or hotels. To support the automatic creation of such bundles, the basic products must have well-defined interfaces with respect to consumer needs, prices, and distribution channels in order to ease information exchange among the suppliers. But information is also vital for the consumer side to assist in decision-making. From this it follows that the tourism industry is an information business. As such, its products have specific characteristics [96]. The tourism product is:

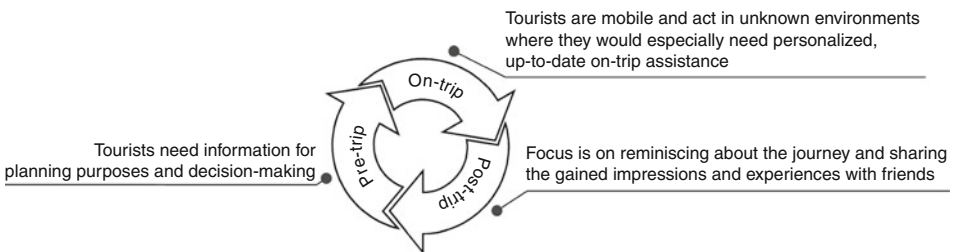
- Perishable, that is, it is time critical and cannot be stored for later usage. A hotel room not sold for a night or an airplane seat being empty represents a lost income. Suppliers are in a risky situation which can be reduced through efficient communication between the different stakeholders and with the consumers.
- A confidence good. Tourists have to leave their known environment and move to another place to consume the product. In this way, the tourist cannot assess the properties of the tourism product in advance. This can be achieved only in the consumption phase. Therefore, decision-making and consumption are separated in terms of time and space. In this situation, information is vital in order to close the gap and deliver that kind of information that helps the consumer to make a choice.

To cope with this situation, tourists usually spend much time in organizing their tour before starting their journey. The result of this planning phase is a personal trip plan that contains an “optimal” schedule of the planned activities.

19.3.2 The Tourist Life Cycle

Even the best trip plan cannot protect tourists from having to face spontaneous, unexpected situations during the trip. Attractions can be temporarily closed, restaurants can change their special offers on a weekly basis, or open-air concerts can be canceled due to bad weather conditions. To counteract such situations, tourists require up-to-date and trustworthy information not only during the journey, but during the whole tourist life cycle shown in ▶ Fig. 19.23, which consists of the phases pre-trip, on-trip, and post-trip. However, the process of selection, configuration, and consumption of tourist information services during these phases is a complex task for the tourist [96].

In the pre-trip phase, tourists need information for planning purposes and decision-making. In the post-trip phase, the focus is on reminiscing about the journey and sharing the gained impressions and experiences with friends. Metasearch and booking engines, destination portals with recommendation systems, and travel communities support tourists mainly in these two phases. In the project *Reisewissen* (<http://reisewissen.ag-nbi.de/>), a hotel recommendation engine has been developed that exploits Semantic Web technology to enhance the quality of the hotel search process. User requirements are semantically matched against hotel resources, resulting in a ranked list of suitable hotels. TrustYou (<http://www.trustyou.com/>) is a semantic hotel search engine, which recommends hotels based on user reviews, which are aggregated from different platforms such as Trip Advisor, Expedia, or Qype. The user reviews are examined through linguistic analysis (negative, neutral, and positive comments can be distinguished) and annotated semantically. In this way, a search query for Barcelona hotels at the beach for good value thus also finds hotels described with cheap rate, incredible rate, as well as beachside location or a great beach. For a specific hotel result, all positive and negative comments gathered from the different reviews are presented in aggregated form to the user.



■ Fig. 19.23

Tourist life cycle

In contrast to the pre-trip phase, tourists are mobile and act in unknown environments in the on-trip phase. There, they would need personalized, up-to-date on-trip assistance in the form of information about accommodation, points of interest (POIs) (e.g., environmental and landscape attractions or gastronomy), flight delays, events weather forecasts, news, or safety issues. Mobile tourist services that can be used independently of temporal and spatial constraints and that are accessed through a mobile handset, may address these issues. In order to prevent cognitive information overload of the mobile user and provide only relevant information, these services should sense and react to the current situation of the tourist, resulting in an increase of the tourist's satisfaction of experiencing a relaxed sightseeing trip.

The results from several surveys [18, 28, 38] show that mobile tourist guides rarely consider information generated by tourists in the pre-trip phase. In this way, they (a) do not incorporate existing tourist profiles (e.g., profile of community member), (b) do not exploit knowledge extracted from the personal trip plan, and (c) do not know the services the tourist is interested in and how these services should be delivered to fit the tourists' requirements and current situation. In all those shortcomings, semantics may play a crucial role.

In the following, trip planning systems are depicted as a reference example for a B2C application in the tourism domain and examined in detail. For that, the following section outlines five issues when constructing such systems. Subsequently, three relevant prototypes are introduced, before each of the issues is discussed in detail. Thereby, it is described how the systems tackle the previously mentioned issues.

19.3.3 Trip Planning: Relevant Issues

A system that allows for creating personalized trip plans and delivering relevant up-to-date information during their journey would certainly enhance the tourists' satisfaction with the planning process and the travel itself. In the ideal case, the entire stay at a destination is planned in space and time. Semantic technologies can help to match between the offerings from the suppliers and the tourists travel preferences. Besides issues concerning the modeling of user preferences or building a destination model, the construction of such a trip planning system also raises certain optimization problems. The overall objective is not only to minimize the travel time between certain POIs. Instead, it seeks to maximize the trip satisfaction of the tourist with respect to a set of criteria, including available travel budget, opening hours, relevance of POIs, time or weather conditions. Hence, the development of a system for providing context-aware, personalized trip plans poses several issues.

The first issue is the *integration of travel information* from heterogeneous touristic data sources. In each of the three trip phases, tourists have varying information needs that have to be satisfied by tourist services. Such services provide travel-related information about hotels, flights, tourist attractions, events and activities, public transport, car rentals, weather forecasts or geospatial information in the form of maps, or routing advice.

In the ideal case, these services cover the whole value chain of tourism, which consists of the phases information/booking, transport, accommodation, and destination/information. In order to (semi)automatically integrate these information sources, an information model has to be employed that structures and classifies the available information in order to enhance information exchange and bundling. This can be achieved by using ontologies, which provide structured (domain) vocabularies, and which are the basis for semantically annotated Web Services.

The second issue deals with the personalization of tourist services. Key for the provision of customized, user-tailored information is the user profile. A system that has no a priori knowledge about the tourist has little chance to offer personalized information. It would need too much interaction to extrapolate the tourist's interests in real time. The main goal is to build a model of the interests and preferences of the user in the pre-trip phase and keep it up-to-date while the tourist is on the trip.

The third issue is the matching of tourist profiles against tourist attractions, which is essential for the creation of personalized trip plans. If the interest profile of the tourist matches the characteristics of a certain attraction, the attraction contributes to the tourist's satisfaction and thus should be recommended to the tourist. Thereby, both the tourist profile and the touristic resources offered by service agents have to be described in an ontological form. The two semantic descriptions are then intersected by the matchmaking algorithm to examine whether they share similar structures.

The fourth issue deals with the selection of a suitable trip planning algorithm in order to generate a route that maximizes tourist satisfaction, but at the same time taking certain constraints such as opening hours of the attractions or trip length into account. Such problems are known as combinatorial optimization problems. A well-known example is the Traveling Salesman Problem (TSP), where the task is to visit every city once and return home covering the shortest distance. Since this problem is NP-complete, heuristic techniques such as local search algorithms can be used to obtain good results in a reasonable time.

The fifth issue tackles the ability of gathering contextual information and performing context-based reasoning. Tourism services have to be context-aware, not only to provide tailored information but also to cope with unexpected situations. Referring to [39], context in the tourism domain can be seen as any relevant information that characterizes the situation of a visitor. This includes information about the tourists themselves, information about their environment as well as information about their objective and planned activities.

19.3.3.1 Trip Planning Applications

Several applications have been developed that provide customized tours for tourists. Some of them are limited to assist tourists in the pre-trip phase by supporting the decision-making process and proposing a trip plan. Others focus on giving support during the on-trip phase, are therefore context-aware and are thus able to provide information more

tailored to the current situation of the tourist. In the following, three trip planning approaches are introduced.

CRUZAR [50] is a Web application that builds customized routes for visitors of the city of Zaragoza in Spain. It exploits expert knowledge in the form of rules and ontologies in order to match visitor profiles against semantic descriptions of tourism data such as historical buildings, museums, events, or parks. A planning algorithm organizes the most relevant sights into an optimal tour.

SPETA [21], short for social pervasive eTourism advisor, is a recommender system that simulates a real tour guide by providing recommender services that are based on the user's current location, preferences, as well as the history of past locations.

The Dynamic Tour Guide (DTG) described in [29] is one of the first mobile tourist guides that calculates personal tours on-the-fly. Tourists can elect their preferences and trip time duration, whereupon the system calculates a customized route, taking into account user knowledge as well as time constraints. During the walk, the system senses the location of the tourist and can adapt the route dynamically if the tourist needs more time at a site than expected.

Issue 1: Tourism Ontologies

The access and exchange of touristic information can be facilitated by describing the semantics of the data in a machine-processable way. Recently, there is a proliferation of ontologies that have been developed in the area of eTourism either by industry, academia, or within collaborative projects. In the following, existing tourism ontologies are introduced.

QALL-ME [61] is an EU-funded project that aims at establishing a shared infrastructure for multilingual and multimodal question answering in the tourism domain. Thereby, it allows users to pose natural questions in different languages using a variety of input devices and returns a list of answers in the most appropriate modality. The QALL-ME ontology provides a conceptualized description of several aspects of the tourism domain, including tourism destinations, tourism sites, tourism events, and transportation. It contains 122 classes and 107 properties that indicate the relationships among the classes. QALL-ME is encoded in the ontology language OWL-DL.

The Harmonise [19] ontology, initially developed within the Harmonise project, is now the central element within the HarMoNET (Harmonisation Network for the Exchange of Travel and Tourism Information) that aims to create an international network for harmonization (the reconciliation process between heterogeneous sets of data) and data exchange in the tourism industry. The ontology focuses on two sub-domains of the tourism domain, namely events (e.g., conferences, sport) and accommodation (private rooms, hotels, guesthouses), modeled in the language RDFS. Members of this network can share data by mapping their specific data model to the Harmonise ontology, which acts as the central data model of the network. The mapping proceeds at the site of each individual member, since there is a proprietary mapping between the member's legacy system and the Harmonise ontology.

The Hi-Touch ontology [51] was developed during the IST/CRAFT European Program Hi-Touch, which aimed at establishing Semantic Web methodologies and tools for

intra-European sustainable tourism. The goal was to formalize knowledge on travelers' expectations and to propose customized tourism products. The ontology was mainly developed by Mondeca and is encoded in the ontology language OWL. The ontology classifies tourist objects, which are linked together in a network by semantic relationships. The semantic network is provided by a Topic Map. The top-level classes of the Hi-Touch ontology are documents (any kind of documentation about a tourism product), objects (the tourism objects themselves), and publication (a document created from the results of a query, e.g., a PDF document). The tourism objects can be further indexed by keywords using the thesaurus on tourism and leisure activities by the World Tourism Organisation. This standard terminology ensures the consistency of the tourism resources categorization managed on distributed databases and enables semantic query functionalities. The DERI eTourism ontology [69] was developed by STI Innsbruck. The ontology focuses on the description of accommodations and infrastructure and enables a user, who queries a tourism portal to find a package of relevant accommodations and infrastructure. The Travel Agent Game in Agentcities (TAGA) [81] is an agent framework for simulating the global travel market on the Web. The TAGA ontology defines travel concepts such as itineraries, customers, travel services, and service reservations as well as different types of auctions. The GETESS (German Text Exploitation and Search System) [78] project focused on retrieving information from touristic websites. This information is semantically interpreted and can be queried by the user through natural language processing techniques. The EON Travelling Ontology was developed by the Institut National de l'Audiovisuel in France. It describes tourism concepts that are divided into temporal entities (e.g., reservations) and spatial entities, which further comprise dynamic artifacts (e.g., means of transportation) as well as static artifacts which comprise town sights or lodging facilities.

CRUZAR's ontology is based on the upper-ontology DOLCE in order to model visitor's profiles, travel routes, and POIs. To describe POIs, it further reuses properties from the Dublin Core, FOAF, and SKOS-Core. SPETA exploits concepts from the eTourism ontology [14] in order to describe tourist services. In addition, it links to concepts from DBPEDIA and YAGO to describe concepts such as attractions or activities and FOAF to describe social links of tourists. DTG's ontology is built leveraging some existing taxonomies from DAML (<http://www.daml.org/ontologies/keyword.html>) and GETTY (<http://www.getty.edu/>). Due to the heterogeneity of the tourism sector, the process of developing and maintaining a single tourism ontology that covers the whole tourism market, including geographical-, temporal-, and user-related information would be very tedious and would require an agreement of the shared vocabulary between the different tourism organizations. Hence, in order to cover the semantic space of the tourism domain and to facilitate interoperability between the different tourism services, a bundle of ontologies may be required as proposed in [9].

Issue 2: Personalization

As tourists have individual preferences, tourist profiling plays an essential role in the provision of personalized travel information. Some preferences of the tourist are rather

static and do not change frequently, such as the age of the tourist, the language spoken, or the food habits. These kinds of data are usually inserted explicitly by the tourists. A questionnaire based on ontology-driven queries can be used in order to elicit information concerning the user. However, inputting personal data into the system is a tedious task for tourists and also involves privacy concerns. Experiments have shown that users faced with more than four prompts for information from a query system tend to give up using the system [100]. Besides, some preferences might be just described at a very high level, which may result in imprecise travel suggestions. For example, tourists may state that they are interested in museums without referring to the category of the museum. In this situation, the system cannot infer whether it should propose to visit a technical or a cultural museum. During the trip, as soon as tourists explore the destination and experience some activities, some preferences as their interests might slightly change and need to be updated. Other factors that influence the situation of the tourist such as the current location and time or the weather conditions change on a regular basis and have to be obtained by the system continuously, with a minimum of user interaction. Otherwise, the tourist would be annoyed and would refuse to reuse the system. Therefore, another approach is to update the profile implicitly by the system through observing the behavior of the tourist and his interaction.

The CRUZAR application obtains the tourists' profiles through a Web-based form. In order to generate a basic trip route, just the travel dates are needed. As the tourist enters more details related to his/her profile, the better can the route be tailored to the tourist's actual needs and requirements. For the customization process no personal information is required and therefore no privacy issues have to be tackled. Tourists can specify the type of travel (e.g., work or tourism), whether they travel alone or in group, and their main tourist interests with respect to their preferred artistic styles or the activities they would like to do at the destination. Whereas the latter has direct impact on the likelihood that certain POIs are interested for the tourist (e.g., if the tourist is interested in the architectural style baroque, buildings with this style are more likely to be included in the trip route), the former influences the generation of the overall trip plan. For example, a family with children on a leisure trip has other demands than a tourist on a business trip. CRUZAR allows tourists to not only specify positive interests but lets them also express dislikes toward certain activities or types of attractions, whose relevance to be part of the route is then lowered accordingly.

SPETA uses two ways to build the user profile. First, it is constructed based on explicit interaction with the user. In addition, social networks are exploited to add further information about the user such as his/her music preferences. Second, the user profile is refined based on the user's feedback while using the system. Based on the type of the museums that the user always visits, his interests can be updated accordingly.

The Dynamic Tour Guide (DTG) uses the interest profile, a start and end point, and a given time period in order to compute a personal tour. In [37], a field study is presented that evaluates different methods for the elicitation of tourist preferences, comprising a hierarchical structure of the ontology using a tree view, using small iconic images to present the levels of the tree and a presentation of just the top-level categories.

Here a novel approach to model the tourist profile is introduced. It is based on a more abstract level and leverages the concept of tourist types, which classifies tourists into different categories according to their needs. Vogt and Fesenmaier [92] propose a model of information needs of tourists forming a categorization of different types of needs, including functional, recreational, as well as esthetic needs. Such needs, wants, behavior, and expectations of tourists can be further classified into tourist types. Gibson and Yiannakis propose in [23] a set of 15 different tourist types such as action seeker, active sport tourist, or independent mass tourist. Tourists typically engage in a variety of these types, whereby types differ from each other with respect to a different set of interest. As described in [23], the interests of the thrill seeker can be described in terms of statements such as “interested in risky, exhilarating activities which provide emotional highs for the participant.” Berger et al. investigated in [11] whether tourist’s preferences can be derived from tourism-related photographs in order to facilitate the process of user profile creation. In this way, tourists can experience a more relaxed way to generate their basic profile.

Issue 3: Matchmaking Process

The need for a matchmaking process arises as soon as tourists would like to get personalized tour suggestions that are tailored to their needs and interests. Thereby, the target of a matchmaking process is finding, for a given tourist profile, those sights that are most attractive to the tourist, which can later be included in the proposed trip route. According to [53], semantic matchmaking is a matchmaking task whereby queries and resources advertisements are expressed with reference to a shared specification of a conceptualization for the knowledge domain at hand, that is, an ontology. If tourist profiles and tourism objects are described in a structured form using ontologies, semantic matchmaking can therefore be applied. Paolucci et al. present in [63] an approach of semantic matching of Web service capabilities. Thereby, they differentiate between four degrees of matching, comprising exact, plug-in, subsumes, and fail. Li and Horrocks [41] extended the work by introducing two new degrees of matching. Di Noia et al. [53] introduce concept abduction and concept contraction and use penalty functions within the matching process.

The matchmaking process of the CRUZAR application is based on the previously mentioned work and follows the following rules. If both tourist profile and resource description share the same features (interests) the score of the resource is unaffected. For each interest in the profile that has no match on the resource side, a penalty function is applied to lower the score of the resource. If the tourist profile includes a dislike, which expresses that the tourist does not want to do a certain activity, but the resource has this feature, then the penalty function is applied as well. Moreover, weights are used in order to express the priority of each feature. A high weight expresses that this feature is required, for example, accessible for handicapped people. If this feature is required but not provided by the resource it will not be included in the trip plan.

SPETA uses knowledge-based filtering techniques in order to personalize the services. As both user and services concepts are structured in an ontology, feature-based similarity algorithms can be applied. The result is a matching score between 0 and 1 that depicts the

similarity between user profile and service description. A certain threshold is applied to filter out elements under a certain matching value.

In the DTG application, a semantic matching algorithm calculates the degree of similarity between the tourism objects and the tourist profile by evaluating the positions of these concepts in the hierarchical tree of the ontology. If an interest concept matches a concept of a tourism resource (i.e., a tour building block), the tour building block receives a certain amount of matching points. If the interest of the tourist is more specific, less matching points are assigned, depending on the distance between the concepts in the tree. Depending on the amount of interests in the tourist profile, the whole hierarchical structure is rated several times. At the end, the points for each tourist building block are summed up.

The photographic profiler [11] can be used to express the tourist profile by using the tourist-type classification proposed by Gibson and Yiannakis [23]. Thereby, scores are assigned to each tourist type, indicating how much the tourist identifies himself or herself with the according type. If the tourist is an active athlete and dislikes organized, package tours, a high value might be assigned to the tourist-type active sport tourist and a rather low value to the type organized mass tourist. The scores can be used in order to reduce or increase the numerical score of those tourism resources that are related to the tourist type. In the previous example, all sports activities should have a high score.

Instead of using semantic matching, vector-based classifying techniques can be used to identify the tourism resources which might be relevant for a tourist. The tourist profile can be represented as a vector. For example, the tourist type profile can be represented through a 15-dimensional vector, where each element represents a tourist type. In a similar manner, such a tourist-type vector can also be assigned to each tourism resource (e.g., a museum), representing the types of tourist that have visited the tourism resource so far. This can be achieved by a mobile tourist guide that keeps track of the sights that the tourist visits and store the data anonymized in a visitor database. If each tourism resource is thus represented as a vector, the distances between the tourist profile and tourism resources can be calculated based on support vector machines techniques which allow the classification of datasets. In fact, through obtaining such data over time, continuous learning can be applied to refine the vectors according to the dataset. Vector-based matching is independent from the semantic description of the concepts.

Issue 4: Context-Awareness and Unexpected Situations

The prerequisite for realizing the provision of customized services is that an application is aware of its context. For this, the classical user model employed for personalization purposes should be generalized to a context model adding primarily environmental data in terms of time and location, together with device and network capabilities. A detailed overview and evaluation of context-aware tourist information systems can be found in [28]. It shows that most of the evaluated systems are capable of sensing the location of the user and provide services that filter the relevant content based on the current user position. Such services are also known as location-based services, as location is the most

important contextual factor. Context-aware services provide more value-added services as they exploit further contextual resources such as the current time, the user environment, or current weather conditions. In order to cope with unexpected situations, tourism services have to be context-aware [28]. Hence, they require the storage and processing of contextual data in a machine-processable form. A number of context modeling approaches [8] exist that differ from each other with respect to the data structures used for the representation and exchange of contextual information. Ontology-based models are a promising approach for modeling contextual tourism information because of their relatively high expressiveness and reasoning capabilities in order to infer additional higher-level situational context information from low-level sensor data by automatic classification. In addition, they are used to solve inconsistencies of sensor data. For building context-aware applications, different frameworks such as SOCAM (Service-oriented Context-Aware Middle-ware), MobiLife Context Management Framework, or CoBrA (Context Broker Architecture) have been developed. CoBrA is based on the ontology SOUPA (Standard Ontology for Ubiquitous and Pervasive Applications). A part of this vocabulary refers to other ontologies, including the Friend-Of-A-Friend (FOAF) ontology DAML-Time and the spatial ontologies in OpenCyc. The CONtext ONTology (CONON) initiative is divided into a common upper-level and a domain-specific ontology and has been prototyped in SOCAM.

Even the best trip plan cannot protect tourists from having to face spontaneous, unexpected situations and unpredictable events. Flight delays, closed attractions, and bad weather conditions force tourists to completely reschedule their trip plan and look for alternatives. The process of trip rescheduling poses a great dilemma for tourists when they have to do this in a manual way without tool support. One issue is the selection of (new) tourist services, which is very cumbersome. Tourists on the move are likely in unfamiliar destinations and have insufficient travel information regarding existing alternatives. In addition, rescheduling of existing activities or including new activities into the trip plan is a complex task as a satisfactory trip plan has to consider total trip duration, opening hours as well as user-defined priorities of activities. Moreover, activities may not overlap and might be dependent on each other (e.g., if the flight is canceled, other activities have to be canceled as well). Being able to react to unexpected events is a challenge for context-aware trip advisor systems and requires automatic processing and the classification of incoming events. Sen and Ma propose in [75] an approach for event-driven processing which is realized by combining reactive rules with ontologies. Beer et al. [10] describe how to use event-condition-action rules for defining and processing context-aware push messages. Rules are used in order to relate messages to situations, which are pushed to subscribed users in reaction to certain incoming events. The proprietary rule language may be substituted by Reaction RuleML, which is a rule family for reaction rules that can react on occurred events by executing suitable actions.

CRUZAR is offered as a Web-based application, which currently supports tourists only in the pre-trip phase and therefore is not able to sense mobile user context. SPETA

takes into account the user's current location, the time and also the weather forecast, history of visited places, as well as friends' recommendations. The DTG exploits the user location and available time frame in order to recalculate the route in case the user gets behind the schedule.

Issue 5: Trip Planning Algorithm

The task of proposing a personal trip plan can be represented as an Orienteering Problem (OP) [84], which is an extension of the Traveling Salesman Problem (TSP). Unlike the TSP, not every attraction (city) can be visited as the travel time of the tourist is limited. Each sight has a certain score allocated to it. The OP algorithm's objective is to generate a route between a set of attractions within the available time frame while maximizing the total score.

The score of each attraction can be computed by matching the tourist profile with its characteristics. This is the task of the matchmaking process, c.f. Issue 3. Based on its result, a numerical score can be assigned to each attraction. Thereby, an attraction with a high numerical score indicates that it is of high interest for the tourist and should be included in the trip plan. The score level not only depends on the tourist's interests but can be also influenced by other factors, such as the type of trip (business vs. leisure trip), size of the travel group, first time visit, but also on certain characteristics of the sight itself. If the sight is a top attraction or belongs to the world heritage sites, it might be ranked highly. Moreover, tourism experts might have predefined a numerical score for all the sights that are worth being visited within a destination. In addition, the score is also dependent on certain context factors such as weather or time. For example, if the weather is bad, the score of all outdoor-located sights, such as a park, might be reduced. The duration of the visit also plays an important role as each sight might have an associated time frame that is the perfect visiting time. For example, an ancient site should be visited in the morning, if it is not crowded with people and the temperature is not so high whereas a beach may be visited in the afternoon when the sun is shining.

For solving the OP, different heuristic techniques such as local search or genetic algorithms can be used. Extensions of these techniques allow one to solve several objectives concurrently, for example, maximizing the total trip score while minimizing travel time, travel budget, or taking into account opening hours. In the work by Souffriau et al. [77], an iterated local search metaheuristic procedure has been developed to maximize the total score, while keeping the total time below the available time budget. Recently, this algorithm has been adapted to run on a mobile device and to take into account the opening hours of the attractions [76].

CRUZAR uses a planning algorithm in order to calculate the route based on two simplifications. First, all movement is on foot and therefore does not require multimodal transportation planning. Second, time is split into different slots where first the morning and afternoon slot is filled; before lunch, dinner and night activities are organized. SPETA does not use a trip planning algorithm. DTG uses a branch-and-bound algorithm to plan a tour by maximizing the values of the attraction scores within a given time frame.

19.4 Related Resources

19.4.1 Key Articles

19.4.1.1 Business Models

Gordijn, J., Akkermans, H.: Designing and evaluating e-business models. *IEEE Intelligent Systems* 16(4), 11–17 (2001). doi:10.1109/5254.941353 [24]

- The article presents a conceptual modeling approach to eBusiness – called e3-value – that is designed to help define how economic value is created and exchanged within a network of actors. It describes the existing gap between business executives and the IT developers who must create the eBusiness information systems and how the e3-value ontology can help to overcome such limitations.

McCarthy, W.E.: The REA accounting model: A generalized framework for accounting systems in a shared data environment. *The Accounting Review* 57(3), 554–578 (1982) [48]

- This paper provides the basic ideas of the REA ontology designed to be used in a shared data environment where both accountants and non-accountants are interested in maintaining information about the same set of phenomena. Within this paper, William E. McCarthy explains the REA model by using data modeling techniques. The paper demonstrates the early backgrounds of the REA ontology and its alignment with the accounting theory.

Osterwalder, A., Pigneur, Y., Tucci, C.: Clarifying business models: Origins, present, and future of the concept. *Communications of the Association for Information Systems* 16(25), 1–25 (2005) [60]

- This paper surveys the different notions and definitions of the term “business model,” and puts them also into a historical “context”; in fact showing that the term only appears in the late 1990s in the academic literature. They distinguish between business model classifications, ontological or modeling approaches, and distinct specific business models. These different views are discussed and put into context, also summarizing the academic discussion.

19.4.1.2 Business Process Models

van der Aalst, W., ter Hofstede, A., Kiepuszewski, B., Barros, A.: Workflow patterns. *Distributed and Parallel Databases* 14(1), 5–51 (2003). doi:10.1023/A:1022883727209 [1]

- This paper identifies a set of reoccurring control-flow patterns in process models and workflow management systems. The identified workflow patterns allow one to

determine the differences between various workflow languages in terms of their expressiveness. The paper evaluates 15 workflow products and highlights their substantial differences. This work can be considered as a cornerstone in workflow and business process research.

Scheer, A.W.: ARIS - Business Process Modeling. Springer (2000) [73]

- This book introduces the well-known ARIS (Architecture of Integrated Information Systems) approach. ARIS supports the modeling and optimization of business processes for realizing them within application systems. The book presents several real-world examples adopting ARIS for the introduction of workflow systems and standard software solutions.

Russell, N., van der Aalst, W.M., ter Hofstede, A.H., Wohed, P.: On the suitability of UML 2.0 activity diagrams for business process modelling. In: Third Asia-Pacific Conference on Conceptual Modelling (APCCM2006). Australian Computer Society, Inc. (2006) [71]

- This paper evaluates UML 2.0 in terms of its control-flow expressiveness. The analysis is conducted based on the workflow patterns framework (see above). Thereby, it highlights the strengths as well as the weaknesses of UML 2.0 activity diagrams when used for business process modeling.

Quyung, C., Dumas, M., ter Hofstede, A.H., van der Aalst, W.M.: From BPMN process models to BPEL Web Services. In: Proceedings 2006 IEEE International Conference on Web Services (ICWS'06), pp. 285–292. IEEE (2006) [70]

- In this paper, the authors investigate a mapping of BPMN diagrams to BPEL processes. In particular, this paper focuses on the differences between graph-based models (BPMN) and block-structured models (BPEL) and highlights the challenges of such a mapping. Inasmuch, existing techniques for mapping BPMN to BPEL have limitations. The authors propose a new technique that overcomes these limitations.

Huemer, C., Liegl, P., Motal, T., Schuster, R., Zapletal, M.: The development process of the un/cefact modeling methodology. In: ICEC '08: Proceedings of the 10th International Conference on Electronic Commerce, pp. 1–10. ACM, New York, NY, USA (2008) [34]

- This paper was among the first to introduce UN/CEFACTS's Modeling Methodology (UMM) 2.0. UMM is a UML-based approach that focuses on the specific needs of modeling global B2B choreographies. In this paper, the authors introduce the new features of UMM 2.0 in order to overcome limitations of UMM 1.0 that have been identified in real-world projects.

Hofreiter, B., Huemer, C., Liegl, P., Schuster, R., Zapletal, M.: Deriving executable bpeL from umm business transactions. In: Proc. IEEE International Conference on Services Computing SCC 2007, pp. 178–186 (2007). doi:10.1109/SCC.2007.49 [33]

- This work aims to bridge the gap between global choreographies and their local implementations in the area of B2B transactions. Thereby, a mapping is presented between graphical UMM models describing the global perspective on an eBusiness transaction and BPEL processes serving as blueprints of the local implementation on each partner's side.

19.4.1.3 Business Document Models

Hepp, M., Leukel, J., Schmitz, V.: A Quantitative Analysis of Product Categorization Standards: Content, Coverage, and Maintenance of eCl@ss, UN SPSC, eOTD, and the RosettaNet Technical Directory. *Knowledge and Information Systems* 13(1), 77–114 (2006) [32]

- The paper of Hepp et al. investigates four different eBusiness categorization standards and assesses their quality and maturity using a framework of metrics. Thereby, the authors go beyond existing research work in this area, where the focus is usually on the architecture and structure of the categorization standards, rather than on their actual content. The result of the analysis shows the strengths and weaknesses of the different approaches, and provides a good starting point for eBusiness categorization standards.

Liegl, P.: Conceptual Business Document Modeling using UN/CEFACT's Core Components. In: *Proceedings of the 6th Asia-Pacific Conference on Conceptual Modeling (APCCM2009)*. Australian Computer Society (2009) [42]

- This article introduces UN/CEFACT's Core Components as a method of choice for modeling the exchanged business document information in an interorganizational business process. Thereby, the author first elaborates on weaknesses of current business document approaches and introduces a UML Profile for Core Components, helping to describe Core Components on a conceptual, model-based level. Finally, the derivation of XML Schema artifacts from conceptual Core Component models is introduced. This paper provides a good starting point to get familiar with the Core Component terminology.

Liegl, P., Huemer, C., Zapletal, M.: Towards a global business document reference ontology. In: *Proceedings of the Third International Conference on Semantic Computing (ICSC2009)*, September 14–16, Berkeley, CA, USA (2009) [43]

- This article further investigates UN/CEFACT's Core Components by providing an ontological reference representation of the Core Component methodology using Web Ontology Language (OWL). The goal of the OWL representation is to facilitate the mapping of existing business document standard definitions onto a common Core Component representation. Exemplarily, a mapping of UBL and OAGi is shown on a conceptual level.

19.4.1.4 eTourism

Staab, S., Werthner, H., Ricci, F., Zipf, A., Gretzel, U., Fesenmaier, D.R., Paris, C., Knoblock, C.A.: Intelligent systems for tourism. *IEEE Intelligent Systems* 17(6), 53–64 (2002) [79]

- This article examines research issues of (future) eTourism systems that support travelers during the full tourist life cycle. Key research topics are discussed, including travel planning systems that help tourists to plan their trips, recommendation systems that support travelers in their decision-making as well as context-aware, location-based services that support tourists while being on the move.

Buhalis, D.: *eTourism: Information technology for strategic tourism management*. Prentice-Hall (2003) [13]

- This book examines the impact of the Information Communication Technology (ICT) on the tourism industry. In the first part, it adopts a strategic management and marketing perspective to explore the relationship between ICT and tourism in general. In the second part, specific tourism sectors are described in detail, thereby focusing on the use of ICT in their strategic and tactical management.

Gretzel, U., Mitsche, N., Hwang, Y., Fesenmaier, D.R.: Tell me who you are and I will tell you where to go: Use of Travel Personalities in Destination Recommendation Systems. *Information Technology & Tourism* 7, 3–12 (2004) [27]

- This article explores whether travel personality categories are a suitable means to classify tourists and can be used as a tool within destination recommendation systems to capture user preferences in a fast way. The results of a comprehensive user survey indicate that travel personalities can indeed be matched with certain travel behavior, and thus be used to provide personalized recommendations about destinations.

Werthner, H., Klein, S.: *Information Technology and Tourism - A Challenging Relationship*. Springer (1999) [96]

- This paper outlines the importance of the travel and tourism industry as one of the leading applications in the business-to-consumer eCommerce. Besides explaining the structural view of the eTourism market, it introduces the tourist life cycle and illustrates how the Web is changing not only the needs of consumers but also leading to an evolution of the market, affecting each market player within the entire tourism value chain.

Werthner, H., Ricci, F.: E-commerce and tourism. *Communications of the ACM* 47(12), 101–105 (2005) [97]

- As tourism is regarded as one of the most successful domains of electronic commerce, this book synthesizes and analyzes the current situation, trying to set the stage for future research. In order to provide a coherent picture, the work is located within a

triangle of tourism research, namely information technology and Computer Science, as well as management science. Concerning these scientific fields, different perspectives are pursued and integrated: a market and industry perspective, looking at trends in the tourism industry, an analysis of the value chain and its redesign induced by modern information and communications technologies, a discussion of organizational impacts and the implications on management strategies, focusing on a business (network) redesign.

19.4.2 Relevant Ontologies

Andersson, B., Bergholtz, M., Edirisuriya, A., Ilayperuma, T., Johannesson, P., Gordijn, J., Grégoire, B., Schmitt, M., Dubois, E., Abels, S., Hahn, A., Wangler, B., Weigand, H.: Towards a reference ontology for business models. In: ER, pp. 482–496 (2006) [6]

- In this paper Andersson et al. investigated the semantic overlap between the three major business model ontologies REA, BMO, and e3-value. Furthermore, they extended the original approaches addressing specific needs for the transfer of resources between business partners. As a result they were able to construct a reference ontology serving as a mapping tool closing the semantic distance between the different business model ontologies. They concluded the paper by introducing certain mapping rules enabling the interoperability between the business model ontologies. Summarized, the paper provides a good starting point to get familiar with the major concepts and semantics of the business model domain.

Hepp, M.: GoodRelations: An Ontology for Describing Web Offers (2008). <http://www.heppnetz.de/projects/goodrelations/primer/> [31]

- In this paper Hepp introduced the GoodRelations ontology as a semantic approach for capturing the essential needs of typical eCommerce scenarios in the commodity segment. The implemented ontology is based on an intensive elaboration using competency questions to investigate the problem domain together with domain experts and stakeholders. Furthermore, the paper discusses existing implementation technologies relevant for implementing the GoodRelations ontology. The paper is well suited for getting an overview about existing approaches and technologies currently available addressing the concise description of products and services in the eCommerce domain.

WSMO Working Group: Web Service Modeling Ontology (WSMO) (2006). D2v1.3, <http://www.wsmo.org/TR/d2/v1.3/> [99]

- The Web Service Modeling Ontology (WSMO) standard defines a formal language for describing various aspects of Semantic Web Services. WSMO has been developed as an ontological description framework based on the Web Service Modeling Framework (WSMF) and comprises design principles for Semantic Web as well as

service-oriented computing. The specification serves as a first starting point for getting familiar with the WSMO ontology. It describes the meta-model structure, principles, and core elements of the WSMO ontology based on the Meta-Object Facilities (MOF).

Fodor, O., Werthner, H.: Harmonise – a Step towards an Interoperable eTourism Marketplace. *International Journal of Electronic Commerce* 9/2 (2005) [19]

- In this article the authors introduce Harmonise as a flexible ontological mapping approach focused on the translation and sharing of XML-based data within a business network. Surveying the needs for a seamless exchange of data schemas in the heterogeneous eTourism domain, the authors propose a reference ontology enabling the seamless transformation of one schema to another. Thereby, they highlight the addressed problem from a socio-organizational as well as technical perspective. Furthermore, the article provides an extensive survey of related academic approaches.

ebSemantics Ontology: <http://www.ebsemantics.net> [58]

- The ebSemantics standardization initiative aims at establishing semantic technologies in the Austrian tourist industry for taking advantage of eCommerce based on Semantic Web. The main focus of ebSemantics is to provide a standardized, structured, and formal description of the tourism domain. The resulting ontologies are a good starting point for investigating relevant structures and related concepts in the domain of tourism.

19.4.3 Key Events

Extended Semantic Web Conference (ESWC)

- The mission of the Extended Semantic Web Conference is to bring together researchers and practitioners dealing with different aspects of semantics on the web. Since 7 years it is a premier international event on Semantic Web research and technologies, with a particular focus on collaborating with other communities and research areas, in which web semantics play an important role – within and outside ICT. It is in its seventh iteration.

IEEE Conference on Electronic Commerce (CEC)

- The IEEE Conference on Commerce and Enterprise Computing results of a merger of the two annual flagship conferences of the IEEE Computer Society Technical Committee on E-Commerce: the IEEE Conference on E-Commerce Technology (CEC) and the IEEE Conference on Enterprise Computing, E-Commerce, and E-Services (EEE). The conference provides a platform for researchers and practitioners interested in theory and practice of technologies to be used in eCommerce and enterprise computing.

IEEE Conference on Services Computing (SCC)

- Since 2004, the International Conference on Services Computing (SCC) has provided a platform for practitioners to present the latest advances in services science. Services account for a major part of the IT industry today. Companies increasingly like to focus on their core expertise area and use IT services to address all their peripheral needs. Services Computing is a new science which aims to study and better understand the foundations of this highly popular services industry. It covers the science and technology of leveraging computing and information technology to model, create, operate, and manage business services.

Information and Communication Technologies in Tourism (ENTER)

- ENTER offers a unique forum for academic, industry, destination managers, marketers and government representatives to explore the future of information and communication technologies (ICTs) in tourism through research and dynamic dialog within the social network of the International Federation for Information Technology and Travel & Tourism community.

International Conference on eBusiness Engineering (ICEBE)

- ICEBE is a high-quality international forum for researchers and practitioners from different areas of Computer Science and information systems to exchange their latest findings and experiences, as well as to help shape the future of IT-transformed consumers, enterprises, governments, and markets. The conference scope spans the areas of Web, databases, service science, multimedia, information systems, and electronic marketplaces.

International Conference on Electronic Commerce (ICEC)

- The International Conference on Electronic Commerce (ICEC) provides a forum for the scientific research community in eCommerce from all over the world annually. It aims to bring together academics and practitioners to explore new frontiers of electronic businesses – with emphasis on cross-disciplinary research, development, and applications.

International Conference on Semantic Computing (ICSC)

- The IEEE International Conference on Semantic Computing fosters the growth of a new research community. It is an international forum for researchers and practitioners to present research that advances the state of the art and practice of Semantic Computing, as well as identifying emerging research topics and defining the future of the field. Semantic Computing addresses technologies that facilitate the derivation of semantics from content and connecting semantics into knowledge, where “content” may be anything such as video, audio, text, conversation, process, program, device, behavior, etc.

International Conference on the Semantic Web (ISWC)

- The International Semantic Web Conference (ISWC) is the major international forum where the latest research results and technical innovations on all aspects of the Semantic Web are presented. As the Semantic Web is rapidly entering the mainstream, ISWC pays particular attention to showcasing scalable and usable solutions, which bring semantic technologies to Web users in authentic application settings.

Workshop on Business and IT Alignment (BUSITAL)

- Organizations are today becoming more and more dependent on their information systems and other kinds of IT-based support systems to realize their business strategies, build value networks with partners, and manage their resources effectively. But how can organizations ensure that their IT investments are well aligned with the needs of the business? A number of frameworks and methods have been designed to help managers in aligning business and IT. Such alignment is a critical “early stage” activity to understand how information systems contribute to business strategy and to set directions for the development and maintenance processes that follow. Recently, novel methods and techniques based on conceptual and enterprise modeling have been proposed to support mutual alignment between business needs and IT solutions. BUSITAL is a forum for practitioners and researchers that want to explore the benefits, challenges, and solutions of business and IT alignment.

19.5 Future Issues

In this chapter the role of semantics in the field of eBusiness has been reviewed, more specifically in the domains of business-to-business and business-to-consumer eCommerce. First, it has been shown how the B2B layer may be split up into three self-contained layers, namely the management, the administration, and the IT layer. Additionally, methodologies have been introduced which may be used to capture the semantics of the management layer and the administration layer. Furthermore, several approaches for the harmonization of business documents being exchanged between different B2B systems have been listed. As a second prototypical application domain, a business-to-consumer example from the eTourism area has been introduced. By using the tourist life cycle as basis, we have reviewed how semantic technologies can be applied in tourism applications, especially in trip planning systems. On the one hand, semantics ease the integration of data from different heterogeneous sources and provide reasoning capabilities on top of this dataset. On the other hand, semantics can be exploited to match between user interests and supplier offerings to provide personalized offers.

However, this chapter also shows that many issues in the field of eBusiness and eCommerce still remain open, not only in respect to seamless B2B integration or in

B2C eCommerce. In the following we list some of these future issues according to users, suppliers, and markets/networks are being summarized:

- Users
 - Analysis, understanding, and modeling of user (behavior).
 - Support of entire consumer life cycles and all business phases, helping them to define, select, and bundle services. Specifically in the case of eTourism this also relates to the integration of the pre-trip, on-trip, and post-trip phase of the tourist life cycle in order to harmonize the respective information flow.
 - Support of user decision processes, including information search and aggregation, mapping user views and attitudes with supplier views.
- Suppliers
 - Service design, modeling, implementation and delivery; this also includes pricing and service/product bundling.
 - Access to legacy systems.
 - Alignment of value views (business needs) with deployment artifacts.
- Markets/networks
 - Analysis and design of dynamic market and network structures.
 - Formal/ontological business modeling in networks.
 - Dynamic network configurations in heterogeneous and distributed environments.
 - Integration of services, processes and applications across company borders, including coordination and mappings of meanings.
 - Provision of a suitable solution for layered business service registries including trust components.
 - Management of privacy, trust, and security.

As usual in application domains these challenges are not only technical. They also involves equally important organizational, social, and economic approaches. However, on a technical level, semantics will be one of the cornerstones within a bundle of different technologies and methodologies.

19.6 Cross-References

- [eGovernment](#)
- [eScience](#)
- [Knowledge Management in Large Organizations](#)
- [KR and Reasoning on the Semantic Web: OWL](#)
- [Ontologies and the Semantic Web](#)
- [Semantic Annotation and Retrieval: Web of Data](#)
- [Semantic Web Services](#)
- [Social Semantic Web](#)

References

1. van der Aalst, W., ter Hofstede, A., Kiepuszewski, B., Barros, A.: Workflow patterns. *Distrib. Parallel Database*. **14**(1), 5–51 (2003). doi:10.1023/A:1022883727209
2. van der Aalst, W.M.P.: Formalization and verification of event-driven process chains. *Computing Science reports* 98/01. Eindhoven University of Technology, Eindhoven (1998)
3. Akkermans, H.: Intelligent e-business - from technology to value. *IEEE Intell. Syst.* **16**(4), 8–10 (2001)
4. Osterwalder, A.: The business model ontology - a proposition in a design science approach. Ph.D. thesis, University of Lausanne, Lausanne (2004)
5. Alter, S., Ein-Dor, P., Markus, L., Scott, J., Vessey, I.: Does the trend towards e-business call for changes in the fundamental concepts of information systems? A debate. *Commun. AIS* **5**, 1–60 (2001)
6. Andersson, B., Bergholtz, M., Edirisuriya, A., Ilayperuma, T., Johannesson, P., Gordijn, J., Grégoire, B., Schmitt, M., Dubois, E., Abels, S., Hahn, A., Wangler, B., Weigand, H.: Towards a reference ontology for business models. In: *Proceedings of the 25th International Conference on Conceptual Modeling (ER 2006)*, Tucson, pp. 482–496. Springer, Heidelberg (2006)
7. Bakos, Y., Brynjolfsson, E.: Aggregation and Disaggregation of Information Goods: Implications for Bundling, Site Licensing and Micropayment Systems. *Lecture Series in E-commerce*. Springer, Heidelberg (2001)
8. Baldauf, M., Dustdar, S., Rosenberg, F.: A survey on context-aware systems. *Int. J. Ad Hoc Ubiquitous Comput.* **2**(4), 263–277 (2007)
9. Barta, R., Feilmayr, C., Pröll, B., Grün, C., Werthner, H.: Covering the semantic space of tourism: an approach based on modularized ontologies. In: *Proceedings of the First Workshop on Context, Information and Ontologies (CIAO 2009)*, Heraklion, pp. 1–8. ACM, New York (2009). doi:http://doi.acm.org/10.1145/1552262.1552263
10. Beer, T., Fuchs, M., Höpken, W., Rasinger, R., Werthner, H.: CAIPS: a context-aware information push service in tourism. In: Sigala, M., Mich, L., Murphy, J. (eds.) *Proceedings of the 14th International Conference on Information and Communication Technologies in Tourism (ENTER 2007)*, Ljubljana, pp. 129–140. Springer, Heidelberg (2007)
11. Berger, H., Denk, M., Dittenbach, M., Merkl, D., Pesenhofer, A.: Quo vadis homo turisticus? Towards a picture-based tourist profiler. In: Sigala, M., Mich, L., Murphy, J. (eds.) *Proceedings of the 14th International Conference on Information and Communication Technologies in Tourism (ENTER 2007)*, Ljubljana, pp. 87–96. Springer, Heidelberg (2007)
12. Born, M., Drumm, C., Markovic, I., Weber, I.: Super - raising business process management back to the business level. *ERICIM News* **70**, 43–44 (2007)
13. Buhalis, D.: *e-Tourism: Information Technology for Strategic Tourism Management*. Prentice-Hall, London (2003)
14. Cardoso, J.: Developing an OWL ontology for e-Tourism. In: Cardoso, J., Sheth, A. (eds.) *Semantic Web Services, Processes and Applications*, pp. 247–282. Springer, Berlin (2006)
15. Chesbrough, H., Spohrer, J.: A research manifesto for services science. *Commun. ACM* **49**(7), 35–40 (2006)
16. Davis, R., Brabnder, E.: *ARIS Design Platform: Getting Started with BPM*. Springer, London (2007)
17. Dorn, J., Grün, C., Werthner, H., Zapletal, M.: A survey of B2B methodologies and technologies: from business models towards deployment artifacts. In: *Proceedings of 40th Annual Hawaii International Conference on System Sciences (HICSS 2007)*, Waikoloa, Big Island, p. 143a. IEEE Computer Society, Los Alamitos (2007)
18. Eisenhauer, M., Oppermann, R., Schmidt-Belz, B.: Mobile information systems for all. In: *Proceedings of the 10th International Conference on Human Computer Interaction (HCI 2003)*, Amsterdam (2003)
19. Fodor, O., Werthner, H.: Harmonise - a step towards an interoperable e-tourism marketplace. *Int. J. Electron. Commer.* **9**(2), 11–40 (2005)
20. Fox, M.S., Gruninger, M.: Enterprise modeling. *AI Mag.* **19**(3), 109–121 (1998)
21. Garc'a-Crespo, A., Chamizo, J., Rivera, I., Mencke, M., Colomo-Palacios, R., Gómez-Berb'ls, J.M.: Speta: social pervasive e-tourism advisor. *Telemat. Inform.* **26**(3), 306–315 (2009)

22. Geerts, G.L., McCarthy, W.E.: An accounting object infrastructure for knowledge-based enterprise models. *IEEE Intell. Syst.* **14**(4), 89–94 (1999)
23. Gibson, H., Yiannakis, A.: Tourist roles: needs and the lifecycle. *Ann. Tour. Res.* **29**, 358–383 (2002)
24. Gordijn, J., Akkermans, H.: Designing and evaluating e-business models. *IEEE Intell. Syst.* **16**(4), 11–17 (2001). doi:10.1109/5254.941353
25. Gordijn, J., Akkermans, H.: Value based requirements engineering: exploring innovative e-commerce idea. *Requir. Eng. J.* **8**(2), 114–134 (2003)
26. Gordijn, J., Akkermans, H., van Vliet, H.: Business modelling is not process modelling. In: *Proceedings of the Workshops on Conceptual Modeling Approaches for E-Business and the World Wide Web and Conceptual Modeling (ER 2000)*, Salt Lake City. *Lecture Notes in Computer Science*, vol. 1920, pp. 40–51. Springer, Heidelberg (2000)
27. Gretzel, U., Mitsche, N., Hwang, Y., Fesenmaier, D.R.: Tell me who you are and I will tell you where to go: use of travel personalities in destination recommendation systems. *Inf. Technol. Tour.* **7**, 3–12 (2004)
28. Grün, C., Pröll, B., Retschitzegger, W., Schwinger, W.: Context-awareness in mobile tourism guides. In: *Handbook of Research on Mobile Multimedia*, 2nd edn., Information Science Reference (2008)
29. ten Hagen, K., Kramer, R., Hermkes, M., Schumann, B., Mueller, P.: Semantic matching and heuristic search for a dynamic tour guide. In: *Proceedings of the 12th International Conference on Information and Communication Technologies in Tourism*, Innsbruck (2005)
30. Hanson, R.: Long-term growth as a sequence of exponential modes. George Mason University, Fairfax (1998)
31. Hepp, M.: Good relations: an ontology for describing web offers. <http://www.heppnetz.de/projects/goodrelations/primer/> (2008). Accessed 29 Dec 2010
32. Hepp, M., Leukel, J., Schmitz, V.: A quantitative analysis of product categorization standards: content, coverage, and maintenance of eCl@ss, UNSPSC, eOTD, and the RosettaNet technical directory. *Knowl. Inf. Syst.* **13**(1), 77–114 (2006)
33. Hofreiter, B., Huemer, C., Liegl, P., Schuster, R., Zapletal, M.: Deriving executable BPEL from UMM business transactions. In: *Proceedings of IEEE International Conference on Services Computing (SCC 2007)*, Salt Lake City, pp. 178–186 (2007). doi:10.1109/SCC.2007.49
34. Huemer, C., Liegl, P., Motal, T., Schuster, R., Zapletal, M.: The development process of the UN/CEFACT Modeling Methodology. In: *Proceedings of the 10th International Conference on Electronic Commerce (ICEC 2008)*, Innsbruck, pp. 1–10. ACM, New York (2008)
35. ISO: Open-edi reference model: ISO/IEC JTC 1/SC30 ISO Standard 14662, 2nd edn. (2004)
36. Kalakota, R., Whinston, A.B.: *Frontiers of Electronic Commerce*. Addison-Wesley, Reading (1996)
37. Kramer, R., Modsching, M., Hagen, K.T.: Field study on methods for elicitation of preferences using a mobile digital assistant for a dynamic tour guide. In: *Proceedings of the 2006 ACM Symposium on Applied Computing (SAC 2006)*, Dijon, pp. 997–1001. ACM, New York (2006)
38. Kray, C., Baus, J.: A survey of mobile guides. In: *Workshop on HCI in Mobile Guides (MGUIDES) at Mobile HCI 2003*, Udine (2003)
39. Lamsfus, C., Alzua-Sorzabal, A., Martin, D., Salvador, Z., Usandizaga, A.: Human-centric ontology-based context modeling in tourism. In: *Proceedings of the International Conference on Knowledge Engineering and Ontology Development (KEOD 2009)*, Funchal (2009)
40. Li, H.: XML and industrial standards for electronic commerce. *Knowl. Inf. Syst.* **2**(4), 487–497 (2000)
41. Li, L., Horrocks, I.: A software framework for matchmaking based on semantic web technology. In: *Proceedings of the 12th International Conference on World Wide Web (WWW 2003)*, Budapest, pp. 331–339. ACM, New York (2003)
42. Liegl, P.: Conceptual business document modeling using UN/CEFACT's core components. In: *Proceedings of the Sixth Asia-Pacific Conference on Conceptual Modeling (APCCM 2009)* Wellington. Australian Computer Society, Sydney (2009)
43. Liegl, P., Huemer, C., Zapletal, M.: Towards a global business document reference ontology. In: *Proceedings of the Third International Conference on Semantic Computing (ICSC 2009)*, Berkeley (2009)
44. Linder, J., Cantrell, S.: *Changing Business Models: Surveying the Landscape*. Accenture Institute for Strategic Change (2000)

45. Magretta, J.: Why business models matter. *Harv. Bus. Rev.* 5, 86–92 (2002)
46. Markovic, I.: Advanced querying and reasoning on business process models. In: *Proceedings of the 11th International Conference on Business Information Systems (BIS 2008)*, Innsbruck, pp. 189–200 (2008)
47. Markovic, I., Karrenbrock, M.: Semantic web service discovery for business process models. In: *Proceedings of the International Workshop on Human-Friendly Service Description, Discovery and Matchmaking (Hf-SDDM 2007) at WISE 2007 Workshop*, Nancy, pp. 272–283 (2007)
48. McCarthy, W.: The REA accounting model: a generalized framework for accounting systems in a shared data environment. *Account. Rev.* 57(3), 554–578 (1982)
49. Missikoff, B.K.: *An Approach to the Definition of a Core Enterprise Ontology: CEO* (2001)
50. Mnguez, I., Berrueta, D., Polo, L.: CRUZAR: an application of semantic matchmaking to e-Tourism. In: *Cases on Semantic Interoperability for Information Systems Integration: Practices and Applications*. Information Science Reference (2009)
51. Mondeca: *Semantic web methodologies and tools for intra-European sustainable tourism*. Technical report, Mondeca (2004)
52. Nitzsche, J., Norton, B.: Ontology-based data mediation in BPEL. In: *Advances in Semantics for Web Services Workshop (Semantics4WS 2008)*, Milan (2008)
53. Noia, T.D., Sciascio, E.D., Donini, F.M.: Semantic matchmaking as non monotonic reasoning: a description logic approach. *J. Artif. Intell. Res.* 29, 269–307 (2007)
54. OASIS: Reference model for service oriented architecture: OASIS standard, version 1.0 (2006)
55. OASIS, U.: *ebXML - technical architecture specification*. Organization for the Advancement of Structured Information Standards, United Nations Center for Trade Facilitation and Electronic Business (2001)
56. Object Management Group (OMG): *Business process modeling notation specification, version 1.2* (2009)
57. OECD: *OECD 2001: Business-to-consumer E-commerce statistics* (2001)
58. ebSemantics ontology (2010). <http://www.ebsemantics.net>. Accessed 29 Dec 2010
59. Osterwalder, A., Pigneur, Y.: An e-business model ontology for modeling e-business. In: *Proceedings of the 15th Bled Electronic Commerce Conference, Bled* (2002)
60. Osterwalder, A., Pigneur, Y., Tucci, C.: Clarifying business models: origins, present, and future of the concept. *Commun. Assoc. Inf. Syst.* 16(25), 1–25 (2005)
61. Ou, S., Pekar, V., Orasan, C., Spurk, C., Negri, M.: Development and alignment of a domain-specific ontology for question answering. In: E.L.R.A. (ed.) *Proceedings of the Sixth International Language Resources and Evaluation (LREC 2008)*, Marrakech (2008)
62. Ouvans, C., Dumas, M., ter Hofstede, A., van der Aalst, W.: From BPMN process models to BPEL web services. In: *Proceedings of Fourth International Conference on Web Services (ICWS 2006)*, Chicago, pp. 285–292 (2006). doi:10.1109/ICWS.2006.67
63. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.: Semantic matching of web services capabilities. In: *Proceedings of the First International Semantic Web Conference (ISWC 2002)*, Sardinia. *Lecture Notes in Computer Science*, vol. 2342, pp. 333–347. Springer, Berlin (2002)
64. Papazoglou, M., Ribbers, P.: *E-business: Organizational and Technical Foundations*. Wiley, Chichester (2006)
65. Petrovic, O., Kittl, C., Teksten, R.D.: Developing business models for ebusiness. In: *Proceedings of the International Conference on Electronic Commerce*, Vienna (2001)
66. Philipp, L., Zapletal, M., Pichler, C., Strommer, M.: State-of-the-art in business document standards. In: *Proceedings of the Eighth IEEE International Conference on Industrial Informatics (INDIN 2010)*, Osaka (2010)
67. Pigneur, Y.: Internet-based information systems for e-business. In: *Proceedings of the Swiss Computer Science Conference (SCSC 1999)*, Lausanne (1999)
68. Pigneur, Y., Osterwalder, A., Dubosson-Torbay, M.: e-business model design, classification and measurements. *Thunderbird Int. Bus. Rev.* 44(1), 5–23 (2002)
69. Prantner, K.: *OnTour: The Ontology*. DERI, Innsbruck (2004)
70. Quyang, C., Dumas, M., ter Hofstede, A.H., van der Aalst, W.M.: From BPMN process models to

- BPEL web services. In: Proceedings of 2006 IEEE International Conference on Web Services (ICWS 2006), Chicago, pp. 285–292. IEEE, Los Alamitos (2006)
71. Russell, N., van der Aalst, W.M., ter Hofstede, A.H., Wohed, P.: On the suitability of UML 2.0 activity diagrams for business process modelling. In: Proceedings of the Third Asia Pacific Conference on Conceptual Modelling (APCCM 2006), Hobart. Australian Computer Society, Wellington (2006)
 72. Sawy, O.E.: Redesigning Enterprise Processes for e-Business. McGraw-Hill/Irwin, New York (2000)
 73. Scheer, A.W.: ARIS - Business Process Modeling. Springer, Berlin/Heidelberg (2000)
 74. Schuster, R., Motal, T.: From e3-value to REA: modeling multi-party ebusiness collaborations. In: Proceedings of the IEEE International Conference on Commerce and Enterprise Computing (CEC 2009), Vienna. IEEE Computer Society, Los Alamitos (2009)
 75. Sen, S., Ma, J.: Contextualised event-driven prediction with ontology-based similarity. In: Association for the Advancement of Artificial Intelligence, Standford (2009)
 76. Souffriau, W., Maervoet, J., Vansteenwegen, P., Berghe, G.V., Oudheusden, D.V.: A mobile tourist decision support system for small footprint devices. In: Cabestany, J., Sandoval, F., Prieto, A., Corchado, J.M. (eds.) Proceedings of the 10th International Work-Conference on Artificial Neural Networks (IWANN 2009), Salamanca. Lecture Notes in Computer Science, vol. 5517, pp. 1248–1255. Springer, Heidelberg (2009)
 77. Souffriau, W., Vansteenwegen, P., Vertommen, J., Berghe, G.V., Oudheusden, D.V.: A personalized tourist trip design algorithm for mobile tourist guides. *Appl. Artif. Intell.* **22**(10), 964–985 (2008). doi: <http://dx.doi.org/10.1080/08839510802379626>
 78. Staab, S., Braun, C., Bruder, I., Dsterhft, A., Heuer, A., Klettke, M., Neumann, G., Prager, B., Pretzel, J., Schnurr, H.P., Studer, R., Uszkoreit, H., Wrenger, B.: GETESS - searching the web exploiting German texts. In: Proceedings of the Third International Workshop on Cooperating Information Agents (CIA 1999), Uppsala. Lecture Notes in Artificial Intelligence, vol. 1652. Springer, Berlin Heidelberg (1999)
 79. Staab, S., Werthner, H., Ricci, F., Zipf, A., Gretzel, U., Fesenmaier, D.R., Paris, C., Knoblock, C.A.: Intelligent systems for tourism. *IEEE Intell. Syst.* **17**(6), 53–64 (2002)
 80. Sullivan, J.O., Edmond, D., ter Hofstede, A.H.M.: Service description: a survey of the general nature of services. Technical report, Report FIT-TR-2003-02. Centre for Information Technology Innovation (2002)
 81. TAGA: TAGA OWL ontology. <http://taga.sourceforge.net/owl/index.html> (2010). Accessed 29 Dec 2010
 82. Tapscott, D., Ticoll, D., Lowy, A.: The rise of business webs. *Ubiquity* **1**(3), 2 (2000)
 83. Timmers, P.: Business models for electronic markets. *Electron. Market* **8**(2), 3–8 (1998)
 84. Tsiligirides, T.: Heuristic methods applied to orienteering. *J. Oper. Res. Soc.* **35**, 797–809 (1984)
 85. UN/CEFACT: UN/CEFACT's modeling methodology (UMM), UMM meta model - foundation module (2006)
 86. UN/CEFACT: Core components technical specification 3.0 (2008)
 87. UN/CEFACT: UN/CEFACT's modeling methodology (UMM), UMM meta model 2.0 (2008)
 88. UN/CEFACT: UML profile for core components technical specification 3.0 (2009)
 89. United Nations Center for Trade Facilitation and Electronic Business: Core components technical specification 2.01 - part 8 of the ebXML framework (2003)
 90. Uschold, M., King, M., Moralee, S., Zorgios, Y.: The enterprise ontology. *Knowl. Eng. Rev.* **13**(1), 31–89 (1998)
 91. Vervest, P., Heck, E.V., Preiss, K., Pau, L.: Smart Business Networks. Springer, Berlin (2004)
 92. Vogt, C., Fesenmaier, D.: Expanding the functional information search model. *Ann. Tour. Res.* **25**, 551–578 (1998)
 93. Weaver, A., Vetter, R., Whinston, A., Swigger, K.: The future of e-commerce. *IEEE Comput.* **33**(10), 30–31 (2000)
 94. Weber, I., Markovic, I., Drumm, C.: A conceptual framework for composition in business process management. In: Proceedings of the 10th International Conference on Business Information Systems (BIS 2007), Poznan. Lecture Notes in Computer Science, vol. 4439, pp. 54–66. Springer, Berlin (2007)

95. Werthner, H.: Intelligent systems in travel and tourism. In: Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI 2003), Acapulco (2003)
96. Werthner, H., Klein, S.: Information Technology and Tourism - A Challenging Relationship. Springer, Vienna (1999)
97. Werthner, H., Ricci, F.: E-commerce and tourism. *Commun. ACM* **47**(12), 101–105 (2005)
98. Wigand, R.: Electronic commerce: definition, theory, and context. *Inform. Soc.* **13**(1), 1–16 (1997)
99. WSMO Working Group: Web service modeling ontology (WSMO), D2v1.3. <http://www.wsmo.org/TR/d2/v1.3/> (2006). Accessed 29 Dec 2010
100. Yu, S., Al-Jadir, L., Spaccapietra, S.: Matching user's semantics with data semantics in location-based services. In: Proceedings of the First Workshop on Semantics in Mobile Environments (SME 2005), Aya Napa (2005). In conjunction with MDM 2005
101. Zeithaml, V., Bitner, M., Gremler, D.: Services Marketing. McGraw-Hill/Irwin, New York (2005)



20 eGovernment

Nigel Shadbolt¹ · Kieron O'Hara¹ · Manuel Salvadorés¹ · Harith Alani²

¹University of Southampton, Highfield, Southampton, UK

²The Open University, Milton Keynes, UK

20.1	<i>Scientific and Technical Overview</i>	851
20.1.1	Introduction: The eGovernment Opportunity for the Semantic Web	852
20.1.2	The Challenges of eGovernment	853
20.1.2.1	Specifically Political Problems	854
20.1.2.2	Challenges that can be Addressed by SW Research	855
20.1.3	Meeting the Challenges	858
20.1.3.1	Challenge 1: Knowledge Representation	858
20.1.3.2	Challenge 2: Integration	860
20.1.3.3	Challenge 3: Publishing	863
20.1.3.4	Challenge 4: Search and Discovery	864
20.1.3.5	Challenge 5: Web Services	866
20.1.3.6	Challenge 6: Privacy and Access Control	867
20.1.3.7	Challenge 7: Reengineering and Change Management	869
20.1.3.8	Challenge 8: The Perceived Costs of Implementing Semantic Web Technology	872
20.1.4	Conclusion	874
20.2	<i>Examples: The Release of Public Linked Data in the UK</i>	874
20.2.1	AKTive PSI	876
20.2.1.1	Context	876
20.2.1.2	Public-Sector Datasets	877
20.2.1.3	Ontology Construction	877
20.2.1.4	Generating RDF	879
20.2.1.5	Migrating to the Web of Data	881
20.2.1.6	Mappings	882
20.2.1.7	Mashing up Distributed KBs	882
20.2.1.8	Inconsistencies	883
20.2.2	data.gov.uk	883
20.2.2.1	Introduction and Context: First Steps	885
20.2.2.2	Government URI Structure	887
20.2.2.3	Versioning	888
20.2.2.4	Provenance	888
20.2.2.5	Core data.gov.uk Ontologies	889

20.2.2.6	Linked Datasets	892
20.2.2.7	Linked Data API	894
20.2.2.8	PSI Linked Data Mash-Ups and Applications	898
20.3	<i>Related Resources</i>	899
20.4	<i>Future Issues</i>	901

Abstract: The use of the Semantic Web (SW) in eGovernment is reviewed. The challenges for the introduction of SW technology in eGovernment are surveyed from the point of view both of the SW as a new technology that has yet to reach its full potential, and of eGovernment as a complex digital application with many constraints, competing interests, and drivers, and a large and heterogeneous user base of citizens. The spread of SW technology through eGovernment is reviewed, looking at a number of international initiatives, and it is argued that pragmatic considerations stemming from the institutional context are as important as technical innovation. To illustrate these points, the chapter looks in detail at recent efforts by the UK government to represent and release public-sector information in order to support integration of heterogeneous information sources by both the government and the citizen. Two projects are focused on. AKTive PSI was a proof of concept, in which information was re-represented in RDF and made available against specially created ontologies, adding significant value to previously existing databases. Steps in the management of the project are described, to demonstrate how problems of perception can be overcome with relatively little overhead. Secondly, the data.gov.uk project is discussed, showing the technical means by which it has exploited the growth of the Web of linked data to facilitate re-representation and integration of information from diverse and heterogeneous sources. Drawing on experience in data.gov.uk the policy and organizational challenges of deploying SW capabilities at national scales are discussed as well as the prospects for the future.

This chapter will consider the specific issues pertaining to the application of the Semantic Web (SW) to eGovernment, and look at some of the ways that the Semantic Web community has tried to address them. The first section will examine some of the challenges and opportunities for Semantic Web technologies within eGovernment, and review progress made. The next section will describe a detailed example of the UK's data.gov.uk program to represent government data on the linked data Web. It will also review pilot work that preceded the data.gov.uk work and which was important in providing insights as to the pragmatic deployment of SW technologies in the public sector. A set of resources will be given for further study, before a discussion of likely future directions of research in this area.

20.1 Scientific and Technical Overview

The Semantic Web (SW) is a response to the increasing demand for information in a range of human activities, as storage and processing have become cheaper, while access to digital information resources grows ever wider. As the benefits are being reaped from the World Wide Web of linked documents, the costs of manual processing are also making themselves felt. One particular challenge that the SW is intended to address is that of the heterogeneity of information sources. Information is created by many processes, is represented in different media in formats of varying levels of formality, and is also of varying quality and completeness. Furthermore, the constructs used in representation – the concepts, predicates, relations, etc. – may have various and different interpretations assigned to them across different resources. However, search, retrieval, and inquiry over

the Web, to maximize the utility of the Web as an information resource, should be performable not only within individual resources, but across resources, however heterogeneous in form and quality. From the openness of the Web comes its great value, and hence it is impossible to insist on particular formats, tools, or vocabularies. Information resources must therefore be *interoperable*, allowing sharing, amalgamation, and exchange of information between applications. Search should be accurate across resources. As has been described in the first volume of this handbook, the SW solution is to provide semantics for information, using ontologies, annotations, dedicated representation formalisms, and other aids to interoperability.

One important application area of the SW is that of government, and specifically eGovernment. In this section, the particular challenges facing the use of the SW in eGovernment, and significant work in this area, are outlined.

20.1.1 Introduction: The eGovernment Opportunity for the Semantic Web

Government and administration generally is thirsty for information; it has been said that the unprecedentedly large information demands of the bureaucracies required to oversee early gargantuan building projects (such as the Egyptian pyramids) led to many important developments in writing and urban living. Although governments are powerful enough to insist to some extent on standardization of information provided to them, and to enforce completeness as far as possible, the complexity of modern administrations means that the centralizing forces are outweighed by the need to delegate the work of government into separate departments, which often become information silos. Intra-departmental efficiency is incentivized by the central government, often at the cost of interdepartmental efficiency.

As the size of government has grown, it has become increasingly reliant upon accurate and timely information about its legislative and policy contexts. Whether that information is gathered by governments, or provided by citizens and businesses, the quality of management of that information is vital. This has led to the promotion of eGovernment to manage information and deliver services using information technology (IT) where possible. Using IT should create a number of benefits for government, including the standardization of processes, efficiency of information transfer, and storage and effective search, not to mention a decrease in the costs of information management. There should also be visible benefits for the citizen, including the simplification of the interface with government, an increase in the transparency and therefore accountability of government, the ability to manage one's own case, and the lower taxes that should result from the reduction of the government's costs.

Government departments have a requirement to share and exchange information meaningfully. So-called joined-up government is a notoriously hard problem, and the nature of the issue suggests that SW technologies have an important role to play, allowing interoperability and transparency, integrating and reasoning over heterogeneous information sources, and supporting connections between different layers of government as

well as across departments. Furthermore, the use of semantic models of interactions should allow for the evolution of systems, reuse of software across contexts and departments, and the creation of customer-focused, multi-viewpoint access to services and information, including the personalization of services.

Heterogeneity is a serious issue for governments. They gather information from so many sources, including their many subdepartments and agencies, and also store so much legacy information (most states have records that predate widespread use of digital technologies) that one must expect the information they use to be extremely heterogeneous. But one must also consider that, unlike most businesses or online services, the users of eGovernment are particularly varied, as they will include, potentially, every single occupant of a country, including people with minimal computing skills, people who are unable to speak the main languages of a nation, people who, perhaps through disability, find it extremely hard to communicate, and even people who are not citizens of the country in question (as well as citizens of a country not resident there).

Hence the SW, a technology to address heterogeneity, has a great deal of potential for supporting eGovernment. However, as the SW is yet to reach full fruition, there is much work, research, and implementation to do. Furthermore, as shall be seen in the next section, the eGovernment domain raises a number of specific problems for the implementation of SW technology.

20.1.2 The Challenges of eGovernment

Recent work on eGovernment has shown that interoperability and reengineering problems can interfere seriously with the effectiveness of government services online. In particular, studies have highlighted the need for standards to support interoperability, security, and privacy requirements that stem from the amalgamation of databases and services, and process reengineering to optimize the benefits of shifting governmental services online. Very few countries are making significant progress to transform their administrative processes – for examples of discussions and policy interventions, see [1–3].

Moreover, despite the opportunities that are available, there is reluctance to follow through, as Peristeras et al. describe:

- ▶ Governmental agencies still publish public-sector information (PSI) using a wide variety of nonstandardized and proprietary formats. The sheer volume and wealth of PSI make the potential benefits of reusing, combining, and processing this information quite apparent. However, agencies typically first express reluctance to make their data available, for various cultural, political, and institutional reasons. So, they keep their legacy systems, and the information stored there, fenced and isolated. Even if they decide to move on and free their data, the different data formats, the lack of commonly agreed-upon metadata, and the absence of standardized vocabularies and definitions result in a huge bulk of practically useless data [4].

The representation of this mountain of data is clearly a vital first step toward semantically enabled government services, and would clearly be an important gain in its

own right. In the example system discussed later in this chapter, data.gov.uk, real evidence of progress toward this goal is shown.

Implementation of eGovernment services of all kinds is usually seen as a stage-by-stage process of increasing political and technological sophistication [5]. Layne and Lee, in common with other commentators, set out a four-stage process, of which the first is cataloging, creating an online presence, putting government information on the Web, creating downloadable forms, etc., giving a one-way communication (broadcasting) facility. In the second stage, the internal government information systems are connected with the online interfaces, and citizens can transact with government, make requests, provide information, fill in forms, etc., making communication two-way. The third stage is one of vertical integration, where local systems are linked to higher-level systems with related functionality. The result for the citizen is a “one-stop shop” that appears seamless. Resources of greater value than information can be exchanged. The final stage is horizontal integration, where systems are integrated across functions, services become ubiquitous, and the individual departments of government become increasingly irrelevant from the point of view of the citizen, who just picks up the services he needs from a single portal. In particular, it is the third and fourth stages that are genuinely transformative of government information infrastructure.

However, it is fair to say (a) that very few eGovernment systems have been genuinely transformative [1], and (b) that the application of SW technology in this space has been more in the realm of prototypes or proofs of concept than fully fledged delivered systems or procedures. A different approach, embraced by the USA’s data.gov (<http://www.data.gov/>) and the UK’s data.gov.uk (<http://data.gov.uk>) initiatives, is to make publicly available much of the nonpersonal data upon which public services and government depend. One then encourages a developer community and third-party organizations to use these data and build their own applications using these, thereby guaranteeing user-relevance and often delivering capabilities not envisaged by the government and its departments. This unanticipated reuse was what drove the success of the Web of documents and it is argued that this is what will happen in a Web of Open Government Data [6, 7].

In this section, some of the challenges to SW technology and research in this domain will be surveyed. However, some problems follow from the nature of the administrative and political process, and clearly cannot be addressed from within the SW community.

20.1.2.1 Specifically Political Problems

Because businesses routinely have to perform reengineering of legacy systems, and because they face similar difficulties, it is tempting to treat government as a large business in the analysis of the problem. However, government has many drivers and difficulties of context that businesses do not face: in particular, whereas businesses have the (relatively) straight forward goal of creating value for shareholders within the law, governments need to meet a wide range of targets and have a range of public tasks and duties. Furthermore, different governments need (or want) to meet different targets.

Governments are extremely large operations compared even to multinational firms, and they are rarely as “lean” and efficient as private enterprises. Government bureaucracies are large employers, and so the politics of employment are brought into play. Changes in working conditions (including redundancies) are sometimes difficult for governments to negotiate. They may suffer at the ballot box if they restructure in too radical a way. Even if they do make workers redundant, they still have to support their former employees, via social security or unemployment benefits.

Hence government has special problems with information, reengineering, and change management generally. Although the SW is clearly an important tool in the future for eGovernment, these problems will of course apply to any move to adopt the SW, imposing costs and requiring new capabilities. A business can provide a business case for reengineering its information systems, and show how value is created and profits maximized. Although value for taxpayers’ money is an important factor in government decision-making, it is not the only one. Indeed, in democracies, where a government will typically face its voters within 4 or 5 years, high perceived short-term costs will often outweigh long-term benefits, which may accrue to a government of a different party.

These problems of reengineering information, which often requires wholesale reengineering of government structures and ways of doing business, and the preservation of trust and privacy, mean that the eGovernment area is an especially complex and demanding application for the SW. It is true that governments do have advantages in that they can enforce rules and standards, and their demands for information are usually met, but the most prominent forces acting upon them are political ones that tend to promote inertia rather than radicalism. The generalized benefits of efficient information flow accrue to everyone to a small degree (including to outsiders who do not vote for the government), while the smaller number of losers, who either have to consume more resources to support reengineering, or who may lose their information-processing jobs altogether, suffer relatively greater losses and are incentivized to become a vocal minority opposed to change.

The problems that attach to any kind of technological or administrative change in a democratic polity cannot of course be solved by SW research, although in eGovernment they are always part of the background. The aim of this chapter is to look specifically at how SW technology and SW research can make a difference. Hence the next subsection will consider the specific challenges where technological research can be expected to make a difference.

20.1.2.2 Challenges that can be Addressed by SW Research

It is clear that, despite the opportunities for knowledge management and integration that the SW affords, there are serious challenges to its implementation in the eGovernment domain, and that therefore need to be ameliorated or overcome. However, progress is being made on all fronts, and so despite the length of the list of the challenges, there is still room for optimism. The major challenges can be enumerated as follows.

1. *Representing information.* Governments need to capture information about themselves, and describe themselves to their citizens and interested stakeholders (such as companies competing for government business), yet doing this effectively remains an unmet challenge [4]. SW technologies, particularly ontologies, can support the creation of portals and knowledge maps that allow stakeholders to discover how eGovernment works in a particular context, and to provide an instrument of analysis for improvement of services [8]. On the other hand, models of citizens' requirements may be harder, although Ilgar et al. [9] have argued that the use of emergent semantics, based on the evolution of "folksonomies" as a result of thousands or even millions of interactions between citizen and government, might "circumvent the problem of ontological drift by dynamically tracking the changing ways in which people conceptualize their domain."
2. *Integration of information.* The particular advantage for the SW occurs when interoperability issues have a semantic dimension [10] – in other words, where the lack of interpretation of data causes administrative obstacles. The information resources in eGovernment contexts are a wide set, including data, documents (including multimedia), files for download, transactions, links, services, and user-provided or user-related items (such as credit card details). These will need semantic markup if SW technology is to allow machine understanding of eGovernment interactions. Also, governments are complex entities, and each entity has its own information and descriptive terminologies. Integration will need to operate with these different terminologies to minimize cost and disruption, and top-down approaches for enforcing integration are not suitable for governments [11].
3. *Publishing information.* Governments need to release information to their citizens under freedom of information initiatives that are spreading across the democratic world, and support and facilitate democratic debate. Traditionally, information law has focused either on protecting information of commercial value (copyrights, trade secrets, patents), or on protecting confidentiality of certain relationships within which private information needs to be readily disclosed (e.g., doctor–patient, lawyer–client). However, many writers and activists, and some politicians, have argued that the asymmetry between government and governed in access to information exaggerates asymmetries of power, and if governments are to be democratically held to account, more information needs to be passed back to citizens. Furthermore, in an argument basically borrowed from John Stuart Mill, others have argued that the quality of policy-making and administration would be improved if more people were involved in them, for which they need information.
4. *Search and discovery of information.* Data in most government organizations are stored in many databases in different departments and locations, yet to realize the potential gains from these data it is necessary to bring it together from across government and elsewhere on the Web to achieve synergy and maximize value. However, the sort of federated search and retrieval that is required is known to be a hard problem. Furthermore, without an idea of what information is available, it is difficult for someone in a particular government department to realize what inferences he or she

could potentially make. The development of policy depends on government officials understanding what information was available to the policy-making process.

5. *Web Services.* Government services need to be configured and composed for delivery to citizens with heterogeneous requirements. For the provision of services, SW services promise greater flexibility and controllability than standard Web Services. Although Web Services can be composed quickly and effectively, their syntactic definitions cannot describe a service's functionality precisely, while the description also needs a human to interpret it in order to determine those contexts of application where it will work, together with the required inputs and outputs. There is also a cost in remodeling every time a new service is deployed. Because of the semantics attached to SW services, context and capabilities can be modeled effectively, and service invocation, discovery, composition, and mediation can be automated, while methods for creating SW services exist that allow new methods to be deployed, or old methods repurposed, without rewriting the entire business process [12].
6. *Privacy and access control.* Although governments are now expected to release information, they have at the same time a requirement for preserving the privacy of their citizens and attending to issues of national security, which is naturally a recurring tension. Data are extremely useful both in the commercial and academic worlds as well as government, yet governments must be circumspect about what they release. Overlapping the need for privacy is the requirement for data protection, which is intended to strike a balance between privacy and fair use, focusing on such matters as good information management, security, the quality of democracy and society (requiring the free flow of some information), freedom of information, freedom of expression, regulation of data stores, and the ability of data subjects to inspect and if necessary amend faulty data [13].
7. *Reengineering and change management.* Governments need to manage the effort of reengineering (both of processes and of legacy data), especially in the context of eGovernment where it is inefficient simply to bolt eGovernment technologies, such as one-stop portals, onto preexisting information silos. A seamless front end usually requires well-designed back-end processes. The use of SW technology to develop eGovernment services and to manage information requires the creation of a number of knowledge resources, including ontologies and process models. Such models are potentially very beneficial for the reengineering effort, as they themselves can be used to visualize the system for its designers, and provide a common understanding of processes for the governmental system as a whole. However, their development can undo, or cut across, many years of information-handling practice, requiring the reorganization of information management. There needs to be a willingness to look at reengineering the fundamental workflow of processes. Furthermore, as boundaries between information sources are broken down, the necessity of particular departments taking specific responsibility for gathering certain types of information becomes less. The logic of merging and amalgamating data is to lower the barriers between people and units traditionally kept institutionally separate.

8. *Perceptions of the costs of implementing SW technology.* Cost–benefit analyses are clearly important in “selling” the SW to an organization, but the problem seems to be that the benefits are hard to quantify, whereas the costs – including conversion costs, maintenance costs, organizational restructuring costs, and transaction costs – seem overwhelming [14]. The original *Scientific American* article popularizing the SW in 2001 [15] put forward a futuristic vision of software agents acting on useful information to perform complex tasks for their users. The standards that are a precondition for such large-scale agent-mediated information processing are progressing, and some are now in widespread use, but the early vision will not emerge until SW standards are well-established and used on a global scale [16]. Likewise for eGovernment, the benefit of adopting SW approaches will only become evident once semantics inside as well as outside governments are more widespread. Perceptions abound about the high cost of developing SW technologies, such as building ontologies and converting data into RDF. As Peristeras et al. [4] argue, the natural inertia of standard hierarchical governmental structures can also help push perceptions of costs in an unfavorable direction, and make the benefits seem less tangible.

20.1.3 Meeting the Challenges

In this section, approaches to the eight challenges outlined above will be reviewed, looking at the potential for improvement and actual progress made in SW research projects.

20.1.3.1 Challenge 1: Knowledge Representation

The increasing digitization of government services has led to a plethora of government-sponsored schemes and architectures to represent governmental knowledge. The Federal Enterprise Architecture (FEA) has been developed by the US Office of Management and Budget to provide a common set of terms and practices for the procurement of IT services, enabling a uniform approach to describing and evaluating IT investment and collaboration. FEARMO (<http://web-services.gov/fea-rmo.html>) is an associated reference model ontology consisting of executable specifications in OWL-DL of five reference models focusing on performance, business, service, technical issues, and data. The specific aim of FEARMO is to serve as a catalyst for innovation in SW technologies to provide interoperability.

The FEA contains a data reference model that provides principles for the description, categorization, and sharing of data. In terms of description, the aim is to provide means for an agency to agree data structure and semantics, using logical or conceptual data models to provide metadata. In terms of categorization, taxonomies (which can be in the form of XML topic maps or OWL hierarchies) are used to give additional meaning to data by relating them to the context of their creation and use.

The UK government has also established a metadata standard, eGIF (<http://www.govtalk.gov.uk/schemasstandards/egif.asp>), with an associated metadata standard eGMS (<http://www.govtalk.gov.uk/schemasstandards/metadata.asp>), which defines terms for encoding schemes, thematic categories and relations using a thesaurus approach, and draws on common and well-used standards, in particular Dublin Core. The Integrated Public Sector Vocabulary (IPSV), a structured thesaurus of administrative activities that was set up for use within eGMS, provides most of the semantics.

As part of the data.gov.uk project the UK government has initiated the development of several core ontologies, including the SDMX/SCOVO ontology for statistical data and the organizational ontology to represent government structures. Also ontologies are available for particular sectors – for example, administrative geography, education, and transport. In a parallel effort the W3C eGov Interest Group has started the development of the DCAT (http://www.w3.org/egov/wiki/Data_Catalog_Vocabulary) ontology to represent data assets.

One foundational ontological structure that has been found valuable for eGovernment is the *life event*, a meaningful entity for citizens (e.g., a wedding, the purchase of a house) that links together administrative services, procedures, and requirements. A standard vocabulary of these, general enough to express the particularities of the varied legislation across polities, while specific enough to give a modeler enough expressive power to build meaningful models, has emerged [17]. Life events have been argued as having several advantages with regard to search, service automation, and usability for eGovernment portals [18]. So, for example, the Access-eGov project (<http://www.accessegov.org/acegov/web/uk/>) [19–21] adopted life events to model government processes from the point of view of users, information consumers, citizens, and businesses. This enabled requirement-driven development of semantic structures based on consumers’ needs, and allowed users to browse the eGovernment site structured by life events.

Peristeras et al. [4] argue that one important challenge for eGovernment, and the related concept of eParticipation of citizens in government processes, is to be able to represent Web 2.0 discussions in order to understand the shifts in public opinion and mood (it may be that work to track opinion and bias, as in the Living Knowledge project [22], would be helpful in such a challenge). The authors advocate the use of Linked Open Data and other Web vocabularies, combined with other technologies such as natural language processing and argument representation, to represent and “easily ‘sense’ what a community wants.” This suggestion, however, ignores some of the dangers of such techniques. For instance, the opinion “worm” that is generated in real time by changes in opinion during leadership debates has become not only an indicator of the progress of the debates, but one of the outcomes. The worm becomes a quantified proxy for debating success, and success in turning the worm upward becomes the aim of the debate for its participants. One could expect a similar result if real-time changes of opinion in the blogosphere or Twitter were tracked. To widen the point, representation of information, particularly about opinion, is not always a neutral move, but can also supply politicians with new incentives and new weapons.

20.1.3.2 Challenge 2: Integration

Integration, or interoperability, is an issue that raises itself in the eGovernment domain in a number of areas. Klischewski [23] gives the following types of interoperability, which together add up to a highly complex set of interrelated problems.

- Citizens' understanding of a situation needs to be mapped onto that underlying the website or service they are accessing.
- Citizens are also likely to be accessing other websites and services in relation to the issue at hand (either for advice, or because the issue requires both governmental and nongovernmental support – for instance, moving house needs contact with estate agency and banking services as well as engaging governmental change-of-address, regional tax, capital gains tax, and local government services).
- Support for the citizen may require integration of private and public resources.
- It may also require process management across government organizations and IT services.
- Citizens' requirements in one area may overlap with requirements in other areas. Someone moving house will need to interface with tax authorities, local government, and utilities. Furthermore, this event may cause other entities to seek services, so a company may need to seek a work permit, utilities may find they need to establish some services to the new address, while the local government may also be affected (e.g., in terms of care for children or elderly people) as a result.

Semantic interoperability – ensuring that the meaning of shared information is interpreted in the same way by sender and receiver – sits alongside other types of interoperability therefore, including organizational interoperability (business processes and collaborations understood in the same way across organizations) and technical interoperability. Of course these are linked – Klischewski [24] argues that within a single organization process integration requires more planning and a higher intensity of cooperation and financial investment, but has a higher potential payoff for success; however, when the integration required is across global, open partnerships, process integration is not usually successful. With a large number of more or less independent units (administrative units within a single government, as well as its network of stakeholders), the successful exchange of information across their heterogeneous IT systems and procedures is far more likely than cross-organization process management, which anyway will begin with information integration as a first stage. Devoted interoperability frameworks usually contain technical standards catalogs, which serve to provide basic guidance to departments, as well as help standardize work procured from outside IT suppliers [25]; and with the spread of SW concepts and services, the SW provides new technical options for achieving information integration [24].

It is also important to distinguish between front-office and back-office integration. In the absence of any kind of integration, the user has to access services in the order required to do a job, and so is in effect required to manage the process himself or herself. During back-office integration, services are integrated at the system level, so a user accesses

a service that then invokes other services as it needs them. In front-office integration, the user accesses a personal assistant, which then coordinates services behind the scenes, thereby integrating services at the user level (e.g., generating a plan for the user's current life event) [26].

The key contribution the SW makes to integration and interoperability is to allow a common ground to be reached without forcing a single perspective on all the actors involved [23]. Without a shared basis, for example, in terms of the actual creation or reuse of ontologies, markup languages, and markup methods, integration will not be achieved because of differences in semantic assumptions made by practitioners in, say, law, policy-making, service delivery, ideology, administrative processes, and IT procurement and management, not to mention the fact that in many nations such services have to be marketed and explained in a number of different languages. Weinstein [27] argues that the requirement to unify, link, or otherwise align models and ontologies enables the identification of commonalities and diversity, thereby facilitating policy discussion in such a way as to foster creative negotiation.

The SW's contribution is based on its philosophy and the formalisms that make up the layered diagram. RDF provides machine-readable descriptions of data, while ontologies expressed in OWL give interpretations of the descriptions with a common vocabulary. Rules and ultimately other types of logic will enable systems to reason about the data. When such a cascade of formalisms is in place, it should enable operationalized methods for resolving differences in interpretation over digital data and services despite structural differences and variable underlying assumptions. If two systems use the same underlying ontology (which is quite possible in well-understood domains), then interoperability will be obtained via a straightforward mapping, but even if they do not and interoperability issues reemerge one level up, their different ontologies can be published, and then mapped or merged. Ontologies are also important methods for guiding citizens through difficult areas with well-established vocabulary and principles that have reached a stable consensus (e.g., law), preventing mistakes in query construction, or misinterpreting answers.

In Europe, the European Interoperability Framework (EIF) has been set up to support interoperability in data exchange, and as part of its principles the Semantic Interoperability Centre Europe (SEMIC.EU – <http://www.semic.eu/semic/view/>) has been created by the European Commission. This is a service for the seamless exchange of data, and focuses on the semantic aspects of interoperability, promoting the reuse of assets both syntactic, such as XML schemas, and semantic, such as ontologies, in the eGovernment domain. It has set up an open repository of what is called interoperability assets, and also maintains a quality assurance framework. The EU has also sponsored a number of projects in the eGovernment field.

For instance, the SEEMP project [28] is an EIF-compliant architecture to support interoperability across public and private employment service agencies. Although each service has its own local ontology for describing at the semantic level the services it exposes and the messages it exchanges, because they are all operating in a reasonably standardized domain, these ontologies are pretty similar. SEEMP has developed a single consistent ontology out of the local ones, which it hopes will become a reference standard

for the employment services who should provide mappings between the local and global ontologies. In a similar approach, the BRITE project [29, 30], which aims to build interoperability between business registers across the EU to facilitate cross-border eGovernment services for businesses, links (and is generated from) divergent national ontologies with a high-level domain ontology (HLDO) that acts as the intermediary between the local domain ontologies.

SmartGov is another EU project designed to specify, develop, deploy, and evaluate a knowledge-based platform to generate online transaction services, which can be easily maintained and integrated with legacy systems. The approach has been to store knowledge units (which could be anything including help, best practice guidelines, examples, troubleshooting advice, etc.), which are often short unstructured pieces of text, in a knowledge base whose structure reflects that of a transactional service. Domain-specific transactions use a domain map to link the transaction structure (and hence the knowledge units) with the domain concepts; the map is based on an eGovernment service ontology [31].

Barnickel et al. [32] point out that a global ontology is impractical for at least some eGovernment scenarios because of the international dimension (systems from different states often have to achieve interoperability), and they argue that service composition in a semantic interoperability infrastructure is the way forward. The combination of domain-specific ontologies and upper ontologies for Web Services, such as OWL-S and WSMO, allows SW services to be wrapped around already existing Web Services (services are discussed in more detail below in [Sect. 20.1.3.5](#)). Semantic bridges [33] describe the relations between distinct concepts defined in different ontologies that nevertheless are intuitively close in meaning; Barnickel et al. [32] use a rule language to do this, ensuring that the transformations can be implemented with an inference engine. Creating such bridges requires cooperation and sharing between domain experts familiar with the (local, domain-specific) ontologies being linked. Semiautomatic tools support service composition, by reasoning over semantically described relationships (such as inheritance between concepts) and recommending suitable assignments between output and input parameters of different services.

As an alternative approach [34], bridges disjoint SW applications by using automatic ontology alignment followed by automatic translation of metadata from one application to the other, allowing services to communicate. Given two SW applications to bridge together, aligning their ontologies produces an alignment file that maps terms from one ontology into the other, which can then be used to translate the output and effect specifications of one application into the ontology of other, which then can be measured against the preconditions of the second application.

In the linked data world several services are alleviating the issue of decentralized data integration. Co-reference systems, such as sameAs.org, help to interconnect linked data resources that represent the same concept or object. A simple RESTful API can be used to discover more information about the same thing in different linked databases. Other services like the EnAKTing PSI Backlinking Service (<http://backlinks.psi.enakt.org/>) help the integration of linked data by resolving the problem of foreign URIs [35]. Linked

data approaches argue for lightweight ontologies and do not insist on large-scale overarching ontologies [16], which can be expensive to build and difficult to maintain.

20.1.3.3 Challenge 3: Publishing

Following the severe financial crisis that began in 2008, the administration of President Barack Obama, which took control in January 2009, put in train a package of economic stimulus of unprecedented size. One key aspect of the package was the need to build and retain voter trust given that the stimulus was close to \$1 trillion, that lawmakers had a poor reputation after a series of scandals, and that banking executives were already perceived by voters and consumers as having manipulated previous systems in order to award themselves large bonuses at the cost of profits for the shareholders, and the gross inflation of systemic risk. Hence transparency about the conduct of the stimulus was seen as central. To this end, a website, [recovery.gov](http://www.recovery.gov) (<http://www.recovery.gov/>), was created to:

- ▶ ...feature information on how the Act is working, tools to help you hold the government accountable, and up-to-date data on the expenditure of funds. The site will include information about Federal grant awards and contracts as well as formula grant allocations. Federal agencies will provide data on how they are using the money, and eventually, prime recipients of Federal funding will provide information on how they are using their Federal funds. Interactive graphics have been used to illustrate where the money is going, as well as estimates of how many jobs are being created, and where they are located. And there will be search capability to make it easier for you to track the funds.

The interest of this site is twofold. First of all, it is underpinned with SW technology, representing and linking data using RDF, supporting queries with SPARQL, and so on. It is also using cutting-edge ideas, such as Semantically-Interlinked Online Communities (SIOC), developed at the Digital Enterprise Research Institute at Galway, Ireland, which is intended to support the development of online communities and linking debates by providing an ontology for representing social Web information in conjunction with FOAF [36].

In the United Kingdom, the Office of Public Sector Information (OPSI – <http://www.opsi.gov.uk/>) led the early drive for the release of public-sector information (PSI), with the aim of “understanding the potential of freeing up access, and removing barriers to reuse, [which] lie at the heart of our push to raise awareness of the potential for transforming how the citizen and state interact” [37]. OPSI operates from within the National Archives, and is at the center of information policy in the UK, setting standards, delivering access, and encouraging the reuse of PSI. It has responsibility for the management of much of the UK government’s intellectual property; it is also the regulator of the information-trading activities of public-sector information holders. As such, it has been in a central position to support the release of PSI using SW technology and formalisms to facilitate discovery and sharing. David Pullinger, Digital Policy Director of the Central Office of Information has supported the use of SW formalisms to link data in a national

information infrastructure, allowing reusable information, represented with SW conventions, in decentralized form but including identity management systems to allow personalization, while an important aim of OPSI is to raise awareness of the SW through government [38]. An extended example of the use of SW technology involving OPSI, called AKTive PSI, is described below in the release of UK PSI (cf. also [39]).

Other participants in AKTive PSI, such as Ordnance Survey, have also made use of Semantic Web technology. For example, the OS GeoSemantics team has released an ontology and dataset of 11,000 instances for Administrative Geography, which describes the administrative divisions in the UK – a complex dataset ideally suited to semantic representation [40]. The success of AKTive PSI was one of the factors that led to the development of the UK transparency and open data initiative data.gov.uk, which will be discussed in detail as an example of the use of the linked data paradigm in government information.

In May 2009, the US government's federal Chief Information Officer Vivek Kundra launched the data.gov website to increase public access to valuable machine-readable datasets generated by the federal government. The philosophy of opening data for citizens' use has been influential in this development. At the time of writing, in the 18 months since its launch it has released thousands of datasets containing billions of RDF triples. In June 2009, Tim Berners-Lee and one of the authors of this paper (Shadbolt) was asked to establish a single point of access – data.gov.uk – for UK public data. It too now hosts thousands of datasets and is promoting the use of linked data standards in the UK Government.

This was not the first time SW technologies had been used by the US or UK governments. RDFa was already in use, to a small extent, behind the scenes on the White House website (<http://www.whitehouse.gov/>) – but the set of technologies in use now is comprehensive enough, as one blogger put it “to enable the citizen masher to do their wizardry.” In 2004, experiments began in the UK to evaluate SW technology. Both data.gov and data.gov.uk are recent sites but their performance and the examples they set will be of great importance as early adopters of large-scale use of linked data in the public realm. Both are likely to be bellwethers of the success of SW formalisms and technology for handling public information. They are governed by similar philosophies of releasing open data in such a way as to allow it to be integrated with and linked to other information sources, using lower-level SW technologies appropriate for the Web of linked data. Both initiatives have benefited from top-level political support from Presidents and Prime Ministers.

20.1.3.4 Challenge 4: Search and Discovery

Improved discovery of information is of course an imperative for an eGovernment system. Comte and Leclère [41] argue that using semantic reasoning will be particularly helpful to address issues of lack of interoperability, poor document management, and the absence of intelligent mechanisms. Interoperability has already been discussed above. With respect to document management, eGovernment information systems are often based on database management systems, but the resources with which they have to deal are often unstructured, and may not even be digital. The failure to index information well

enough can also cause it to be effectively lost, as irretrievable. The absence of intelligence and inference is important, say Comte and Leclère, because of the low level of familiarity and expertise of the client, the citizen. They give the example of someone with a request in the legal domain who needs to have not only an answer, but also a reassurance that the answer is complete and correct, that no other relevant information has been missed out, and that information held implicitly in the information source has been rendered explicit. In their system, they rejected the classical database closed-world assumption, and built a portal in which each government information resource is described by metadata using vocabulary from a heavyweight ontology, which can be displayed in networks. A set of protocols and processes allows the importation of data specified in RDF and OWL, and the reasoning uses an AI formalism based on Sowa's conceptual graphs that has an expressivity equivalent to RDFS.

Sidoroff and Hyvönen [42] argue that the application of SW techniques to the problems of content discovery and aggregation in eGovernment portals is highly beneficial, allowing semantic search and dynamic linking between pages. This in turn allows the consolidation of heterogeneous content related to the same information need. They have used SW techniques on the Suomi.fi eGovernment portal based around the idea of life events [17]. This allows the development, for instance, of *compound pages*, which tie together several information sources into a list in a single resource, aggregating information from different sources in a clear way. Explicit logic rules in SWI-Prolog allow the generation of links dynamically, making it possible to link any information resources in the portal that satisfy a linking rule, for example, exploiting similar properties of the resources expressed as metadata, or providing navigation hints based on recommender systems. Search allows content to be classified and viewed along several orthogonal views simultaneously. Ontologies (defining views along dimensions such as life events, topics, location, target audience, etc.) are used to describe Suomi.fi's information items, which are then used to represent the information in taxonomies for the user interface. A top-level view shows the whole subject topic taxonomy, which the other views allow alternative ways of classifying the content; the approach insists that new kinds of view for search can be created easily and the content projected onto them as the user requires.

Peristeras et al. [43] describe a use case and a platform for discovery of eGovernment services based on an OWL representation of the Governance Enterprise Architecture (GEA), which enables the semantic matching of citizens' profiles with formal specifications of available and relevant public services. A similar ontology-driven approach has been described in [44], in the development of a semantically enhanced search system to retrieve statistics, a large and growing space of data. This paper puts forward an intelligent search engine based on modeling electronic catalogs in the EU Combined Nomenclature with OWL. The search technique combines standard keyword-based search of the actual database with a higher-level RDQL (RDF Query Language) query of the ontology. The SAKE project [45, 46] provides a framework and tool workbench for an agile eGovernment change management, knowledge sharing, and knowledge and service creation, detecting change in information sources and work contexts and proactively delivering resources to users.

The hierarchies can be very useful navigational aids for the user as well. Sacco [47] argues that the use of *dynamic taxonomies*, which can adapt dynamically to the user's focus, allowing items to be classified under an arbitrary number of concepts at any level of abstraction, solves many of the problems of search and discovery for citizens who are perhaps only tenuously aware of what they are looking for – the search does not depend on the starting point, and it produces all potentially relevant information (e.g., all services offered to senior citizens) before drilling down to find the particular requirement.

A different principle is in operation in the exploitation of linked data. As part of the data.gov and data.gov.uk, an increasing number of linked data mash-ups have been noticed. As already noted the overarching principle here is that government makes data available and then a large developer community outside government builds the applications. Examples of these will be given in the example application section below.

20.1.3.5 Challenge 5: Web Services

The Web Services field is seen by many as the silver bullet for the SW within eGovernment, especially as the provision of services is a key function of government. Moving services to the Web would provide high availability and facilitate reusability [48].

The composition of government services (both on- and offline) has generally been ad hoc and time consuming, and this issue has also afflicted the field of Web Services as well. Traditional languages for describing operational features of Web Services to enable service composition, such as the Web Services Description Language (WSDL), the Simple Object Access Protocol (SOAP), or Universal Description, Discovery, and Integration (UDDI) have little or no support for the semantic description of services that would allow automatic discovery, selection, and composition, although they could be supplemented by ontologies [48–52]. Chun et al. [50] argue that automation requires rules describing nonfunctional capabilities and properties of services as well as syntactic and semantic descriptions, and organizes its approach around policy rules specifying how the actual administrative unit handles contracts, negotiations, and other pragmatic, contextual aspects. The Web Service Modeling Ontology (WSMO – <http://www.wsmo.org/>) is a conceptual model that can be used to create domain ontologies, as well as for specifying nonfunctional properties of services (such as their cost and quality), describing the goals of the functional capabilities, describing service-related information (such as the functional capabilities, its communication protocols, or its decomposition in terms of other services), and specifying mediators by identifying and removing obstacles to compatibility at the level of data format or underlying processes.

The EU project DIP (Data, Information and Process Integration with Semantic Web Services – <http://dip.semanticweb.org/>) developed an eGovernment use case in close collaboration with Essex County Council, a large local authority in South East England (UK) containing a population of 1.3 million, to deploy real-world SWS-based applications in such a way as to support reuse and composition [12]. In order to provide semantics and step toward the creation of added-value services, DIP adopted WSMO and IRS-III, a tested implementation of this standard [53]. Since government legacy systems are often

isolated – they are not interconnected and/or use distinct technological solutions – the innovations of the DIP approach firstly enabled the data and functionalities provided by existing legacy systems from the involved governmental partners to be exposed as Web Services, which are then semantically annotated and published using SW service infrastructure.

Using SW services with formal descriptions of their semantics, machine interpretation enables discovery by matching formal task descriptions against the descriptions of the services, mediation, and composition [54]. The advantages of this for eGovernment, as set out by Gugliotta et al. are:

- Providing added-value joined-up services by allowing software agents to create interoperating services transparently and automating integration
- Enabling formalization of government business processes, allowing a common understanding and visualization across heterogeneous administrative units, possibly promoting reengineering
- Reducing risk and cost, moving from hard coded services to reusable ones
- Allowing one-stop customer-focused and multiple viewpoint access to services

Gugliotta et al. [55] based one of their scenarios around the “life event” concept, envisaging an active life event portal. WSMO augmented with ontologies based around the “life event” concept is also the basis for projects such as Access-eGov [20], while Wang et al. [56] describe the extension of WSMO to encompass public administration concepts, linking the generic public service object model of the GEA with WSMO to produce WSMO-PA. Further experiences of using the GEA are presented in [43, 57, 58].

20.1.3.6 Challenge 6: Privacy and Access Control

Privacy and trust are essential factors for eGovernment, and the SW would seem to be a valuable tool to promote them, with ontologies providing vocabularies to express privacy and security policies machine-readable, allowing reasoning over them (the variability of people’s attitudes to privacy, and therefore the policies they will endorse, is one major reason why SW technology has great potential here). Where a government possesses large quantities of information, the guarantor of privacy is often what might be termed *practical obscurity*: the phenomenon that information, often paper-based and held in discrete repositories, though theoretically in the hands of governments, is actually not useful because it cannot be found effectively in a timely way [59]. This is particularly true of information that does not exist explicitly in government archives, but could be deduced from information held in two or more other sources. Hence in some polities lack of trust in government, however well-founded, can lead to skepticism regarding the benefits of efficiency, because efficient use of information can lead the citizen to feel that their personal affairs and actions can more easily be scrutinized by government. Hence, when it comes to the citizen trusting the government’s use of IT innovations, privacy issues loom large.

However, there is relatively little work in the field as yet discussing privacy and trust. One can only speculate why there is such a lack. It may be that, in such a complex

field where relatively little progress has been made in the more transformative areas, most effort has been focused on proofs of concept, defining architectures and services that deliver in real-world contexts, with privacy considered as a bolt-on to be addressed in future prototypes that are closer to genuine implementation. For example, Klischewski and Jeenicke [60] explicitly remark that their prototype system for the Hamburg area focused on service functionality, and privacy was a secondary issue not addressed, despite the fact that they focus strongly on requirements analysis. Peristeras et al. [4] argue that, although privacy will inevitably be a problem with linked open government data, “there are many ‘safe’ candidate start-up areas for which information reuse looks quite harmless: for example, data related to geography, statistics, traffic, and public works.” Indeed the data.gov.uk work explicitly focused on nonpersonal public data to avoid issues around personal and private data. Nevertheless, citizens, developers, and governments may discern benefits in combining personal and nonpersonal data held by the state, in which case privacy concerns will need to be addressed.

Medjahed et al. [52] describe the privacy solutions used in WebDG system, an architecture for providing customized SW services in the USA in the domain of benefit collection (WebDG is a generic system, based on WSDL). The benefits area has obvious privacy issues (citizens must disclose their social security number and salary, for instance), and the information is even more sensitive when combined illicitly with other information (e.g., from the tax office, about earned income). They point out that security does not necessarily produce privacy, because one may want one’s information kept private from people operating a secure system (e.g., government employees). WebDG has a three-layer privacy model. User privacy preferences are specified through editable profiles (which can be overridden by government regulations), and WebDG assigns a user a credential on the basis of these. The credential determines access and read/write rights to data objects. Each eGovernment service has its own privacy policy specifying a set of rules applicable to all users, stating the purposes for which the service can use information collected, the length of time, and the conditions of information storage, and specifying how and to whom the information can be released. And data also has associated with it a privacy profile that determines the access views it allows to those who access it. WebDG also contains a series of modules intended for enforcement of policies.

Weitzner et al. [61] argue that attempting to control access to data in a world of digital information, where copying and transmission are virtually costless and the large size of communicating networks means that dissemination can be extremely wide and fast, is fundamentally mistaken and destined to be behind the curve of progress. The authors argue that data use is a more important thing to focus on, and that data access control should be supplemented by legal and technical mechanisms for transparency and accountability, allowing harms to be traced and rectified or compensated. The Policy Aware Web (<http://www.policyawareweb.org/>) is a conception of the SW intended to allow this idea to happen.

Weitzner et al. [61] describe a transparency and accountability architecture TAMI designed to address data misuse in the scenario of government data mining of transport information in order to identify potential terrorists. The architecture needs an inference

engine to support analysis of data and assess compliance with rules of access and use, a truth maintenance system containing proof antecedents from the inference engine, data provenance, and justifications of the antecedents, and a proof generator. This architecture is intended to identify factually incorrect antecedents (e.g., misidentifications of passengers), assess compliance with information sharing rules prior to data transfer, and to check that actions are consistent with information usage rules.

This is an interesting example of the way that technical developments can affect process reengineering. Weitzner et al. [61] first argue that a privacy focus on data use is far more realistic in the current technological climate than a focus on access, and then uses the proposed architecture as an intervention in the public policy agenda. It claims that “policy aware systems bring added focus to policy questions regarding data mining privacy,” and that to realize the promise of transparency and accountability rules, a series of legal issues will have to be resolved – for example, a question such as “under what circumstances, if ever, can inferences generated in one type of profiling system (e.g., anti-terrorism passenger screening) be used to further criminal investigations?” In this way, the SW can be transformative in e-government, by posing and demanding answers to hard questions. This brings the discussion onto the next challenge, about reengineering and change management in general.

20.1.3.7 Challenge 7: Reengineering and Change Management

The issues underlying reengineering should not be underestimated. It is very hard to turn staff-intensive and paper-based systems into automatic digital systems, especially when the reengineering might be entrusted to the very staff whose jobs are under threat from the transformation, and whose incentives are at best mixed. It is also very hard to integrate systems across platforms to provide seamless service for the citizen. Furthermore, the chief driver of change is not pressure from without, but rather consciousness within government of the opportunity costs of not upgrading systems – a notoriously weak driver. As a result, twenty-first century eGovernment systems are often grafted onto nineteenth-century bureaucracies. This locks-in the high costs of integration, and tends to create islands of eGovernment rather than allowing an integrated approach across government.

Stojanovic et al. [62] argue that the use of semantic technologies to describe eGovernment services can improve the management of change in the resolution of either static modification or dynamic modification.

- ▶ Taking into account an enormous number of public services and dependencies between them, as well as the complexity of interpreting and implementing changes in government regulations, the process of reconfiguring the existing legacy systems (the so-called static modification) seems to be quite complex. Indeed, an efficient management system must provide primitives to allow the progressive refinement without rewriting it from scratch, and must guarantee that the new version of the service is syntactically and semantically correct. However, an efficient management system for resolving *static* changes in an e-Government domain does not exist.

The SW, they argue, can provide tools for addressing this problem. This is agreed by Klischewski and Ukena [63], for instance, although their focus on requirements analysis biases their approach toward the creation of SW services in a static regulatory context, which is certainly important, yet ignores the perhaps more frequent situation where the regulatory context is dynamic, and hence methods to deal with dynamic modification are also needed. Peristeras et al. [4] argue that annotations are important in the creation of public knowledge, as “the key to knowledge creation lies in the mobilization and conversion of tacit knowledge,” and suggest that documents be enriched with knowledge drawn from other documents and communities.

Accordingly, Stojanovic et al. [62] add ontologies for understanding the evolution of the system’s ontologies as procedures and regulations change the usage of the system by end users (the ontology is used to structure the usage log) and the life cycle of the system as a whole (which describes information flow and decision-making in public administration). Such a system is intended to locate out-of-date services (i.e., ones that have ceased to be relevant because of changes of regulation, possibly in quite a remote domain), and manage the change and change propagation process – thereby creating a bridge between decision- and policy-making and technical realization. To do this, it is not enough to provide semantics for the relevant services, but all the dependencies between stakeholders that come together to create collaboratively the administrative processes need to be modeled. The management of change, including representing change at the right granularity, and propagating change through the system, is described in [64].

The large-scale model, including models of all the services included, is then used, together with a set of constraints that must be satisfied for all services (included in an ontology to describe the services), to ensure consistency with each other, and with the constraints that together define an ideal for eGovernment services. A typical set of models using such a methodology can be clustered according to whether they are meta-ontologies that define the modeling language for eGovernment services, domain-specific ontologies, or administration ontologies. The meta-ontologies include legal ontologies, organizational ontologies, and the life-cycle ontology, as well as the common device of a life event ontology [65]. Hinkelmann et al. [66] found such a methodology very useful in dealing with the loose federal structure of government in Switzerland.

The constraints include “each service has to have reference to one business rule” (i.e., each service has to be invoked by some process, otherwise it should be deleted), “each service input has to be either the output of another service or specified by the end-user” (this enables changes in one service to propagate to other services that use its outputs), and “if the input of a service subsumes the input of the next service, then its preconditions have to subsume the preconditions of the next one” (preventing nonoptimal configurations in the event of a change, so that for instance if one service has as a precondition that the user is over 18, there is no point having services that use its output having a precondition that the user is over 16). Most of the constraints are domain-independent in this way, although as Stojanovic et al. [65] point out, domain experts are extremely important in this methodology as they possess very good knowledge about the effects of eGovernment in their area, and they understand legislation, the legislative history and context, and the public view of it. The constraints are

required, because changes in administrative processes would not necessarily introduce inconsistencies with the ontologies, or with the services themselves – the constraints are needed for change management. Once there has been a change in administration or regulation, a procedure analyzes the current model to see where there are weak points, if any, signaling this to managers enabling them to produce technical fixes.

Stojanovic et al. [62] argue that such a system would ultimately be able to suggest changes that improve services even where administrative change has not taken place, by being used to monitor eGovernment service execution in the context of citizen comment and complaint about the system. Stojanovic et al. [67] explored this possibility in the FIT project (Fostering self-adaptive eGovernment service Improvement using semantic Technologies), in which the methodology and models they outlined were used to provide a personalized and inclusive (i.e., taking user needs and preferences into account) front office, and a flexible back-office that supports knowledge sharing, best practice, multi-context views, and context-aware service delivery. So-called front-office integration takes precedence over costly back-office integration. With front-office integration, service integration does not require intervention in their implementation, and it is neutral between newly built services and legacy services. Integration need only go as far as is driven by demand, and indeed special purpose customized suites of services can be created for individuals [26]. The system can learn preference rules from citizen interactions, and modifies the service portal according to changes in user requirements and feedback. The FIT ontology, an eGovernment upper ontology, is the basis for navigation and inference across the system.

A similar idea, of using a model distributed about ontologies, including ontologies of change, informs the OntoGov project (<http://www.hsw.fhso.ch/ontogov/>) [65, 68, 69], an EU project that ran between 2003 and 2006 to develop, test, and validate a semantically enriched and ontology-enabled platform to facilitate the management, including composition and reconfiguration, of eGovernment services. This defined a high-level generic ontology for the service life cycle covering all the phases from definition and design through to implementation and reconfiguration to provide the basis for designing lower-level domain ontologies specific to the service offerings of the participating public authorities. It also developed a platform to enable public administrations to model the semantics and processes of eGovernment service offerings at different levels of abstraction, and to enrich the provision of eGovernment services to citizens and businesses with useful metadata. The aim of OntoGov was to bridge the gap between policy-making and the technical realization of eGovernment services, making knowledge explicit and supporting the management of services over their life cycle.

As well as evolution of ontologies, versioning of data is extremely important, and is one of the main issues addressed by data.gov and data.gov.uk. For instance, most of the PSI datasets released by the UK government contain past versions and are liable to change in the future. Regional boundaries and legislation are two simple examples of domains that change regularly, and which are crucial to any eGovernment application. The technical issues behind versioning in linked data and the solution that data.gov.uk is implementing by using RDF graphs will be summarized in the example application section below.

Ultimately, reengineering depends not only on the use of semantic technologies, but also on the *perceptions* of technologists and administrators underlying the management of change. The SW has something of a bad reputation when it comes to implementation issues, and it can be hard to get buy-in from administrators who anticipate a large initial investment cost following a disruptive information management phase. These worries of perception are overblown, and will be discussed in more detail in the next section.

20.1.3.8 Challenge 8: The Perceived Costs of Implementing Semantic Web Technology

As Fishenden et al. [70] argued about the problem of interoperability in eGovernment, “Interoperability is concerned with more than just low-level technical issues. To be successful, interoperability programs need to address a range of issues that span technical, semantic, cultural and organizational interoperability as well as security, confidentiality, data protection, privacy and freedom of information obligations.” Hence there are a number of diverse obstacles to using SW technologies and formalisms to address eGovernment problems. The investment in reengineering may look daunting in terms of initial cost, and possibly even in terms of expected benefit, at least until enough concrete and irrefutable evidence has been amassed that the SW does deliver. Well-placed champions will be important.

This leads to a serious pragmatic issue about how such champions should be deployed, and where they should be within the organization. Is it better to use a “top-down” approach to conversion, engaging a powerful person or administrative body high up in the hierarchy, which prescribes methods for interoperability, determines how many resources will be devoted to the reengineering, and is prepared to provide incentives for change? Or alternatively should a “bottom-up” process allow interoperability to emerge via smaller units at the leaf nodes of the hierarchy engaging in information sharing, perhaps in a series of bilateral arrangements, culminating in the emergence of a *de facto* method, which then can be formalized?

In a complex and fragmented domain such as eGovernment, it is clear that some bottom-up processes will be required [11], because so many different cultures (e.g., formal vs. informal models) and practices will be used. Nevertheless, some kind of top-down pressure will also be required (a) to ensure that those departments reluctant to change still undergo the process, (b) to ensure consistency between approaches and to avoid reinventing the wheel (e.g., by sharing of ontologies), (c) to steer reengineering strategically, and (d) to provide rewards and incentives for good practice, and manuals of best practice. Klischewski [23] argues that the way to reconcile the top-down and bottom-up approaches for producing semantic integration and common ontology acceptance requires three essential steps.

- First, metadata standards such as eGIF or other initiatives drawing on standards such as Dublin Core should define terms for administrative purposes.
- Second, integration should be framed through upper- and lower-domain ontologies, finding common ground on generic concepts and also on elementary concepts, and providing mappings between the often divergent concepts in between.

- Third, intermediate between conceptual schemas acknowledging the importance of each departmental culture, while creating reference ontologies to support mappings between schemes.

Wagner et al. [11] suggest that the complexity of existing eGovernment systems is likely to be an important challenge to the SW in this area, and insist that any approach must incorporate the bottom-up methodology in at least some respects. They propose the design of a two-layer wiki with a semantic layer describing semantic relationships, maintained by a community of users, although as they admit such a methodology raises questions about trust (of the information created) and confidentiality. However, as they maintain, there is a broad trade-off between accuracy and trust on the one hand, and broad and easy participation on the other. The overlaid semantic wiki would identify semantic relationships at the content layer and express them in a separate structure in machine-readable wiki pages. Pages expressing the logic of the eGovernment SW would then allow, for instance, a matching of related pages; question pages, which state questions and link to multiple options, could then be associated with explanation pages that provide explanatory content. This sort of structure could be laid over legacy pages as well as newly generated content, allowing machine interpretation and automatic provision of links.

Nevertheless, as Fishenden et al. argue [70], one needs to look beyond the technical as the spread of SW technology in eGovernment is assessed, whichever problem one is concerned to address. It is probably fair to say that many organizations still view the Semantic Web with some skepticism, and this culture needs to be addressed. It thrives no doubt partly because of a suspicion that administrators are expected to pioneer an approach in which there are few “quick wins.” Moreover, there may be worries about the cost and privacy issues that arise whenever increasing amounts of information are linked into the Web.

Some have produced a reverse pragmatic argument for using semantic technologies in eGovernment, that even if there are high perceived costs, they are an important means to move toward transformation. If the aim is to produce horizontally and vertically integrated eGovernment, a semantic representation of government data, maximizing the use of external data and also releasing government data to the outside world will be important steps toward that goal. On the other hand, as Koné et al. argue [71], the SW cannot do everything. The semantics will help, but culturally, cooperation will still need to be fostered, while organizationally, different administrative environments will still need to be brought together to use, or at least to make reference to, standard reference ontologies. W3C’s interest group on eGovernment has published a working draft on how to publish open data [72], and recommends the following steps: (1) publish data in its raw form if they are structured and can be extracted from the document; (2) create a catalog with documentation of what is available; and (3) convert the data so they are human- and machine-readable, with semantics, metadata, and URIs.

The particular case of small governments has also been studied [73]; such governments are thought to lack the management and reengineering resources to improve semantic interoperability of distributed eGovernment services and resources, partly because of the complex requirements highlighted earlier, where the creation and

publication of information with machine-readable annotations is not a simple process. The level of support required is large, and the initial costs can also be very risky to take on. It follows that quick wins and a lowering of ambition (e.g., not using a single elaborate ontology, but multiple overlapping small-scale ontologies) will be important factors here [74]. Klischewski's work in Schleswig-Holstein, a small German state containing a thousand heterogeneous municipalities, revealed that maintaining up-to-date and standardized information bases locally for use by the state government is hard (small municipalities do not have the workforce), while central databases are inefficient. Therefore any central eGovernment application has to obtain the required information from heterogeneous local sources, and motivating such municipalities to cooperate requires a deep and sympathetic identification of their requirements and constraints, and some transfer of resources downward (e.g., for provision of new methodologies and tools).

Given that an opportunity for the SW to deliver benefits to government has been identified, and that some person or people within an appropriate institution are keen to implement the technology, it is important to understand and counter the common misconceptions held about the SW [74]. Common misconceptions are described in [Table 20.1](#), along with a brief reality check.

20.1.4 Conclusion

The challenges outlined in this section are the sorts of challenge that present themselves to any organization attempting to solve similar problems or grasp similar opportunities for integration and upgraded information management. However, governments feel the difficulties more strongly, as the pressures on them are more acute and diverse. The responses outlined to the challenges above have stressed the pragmatic element, as well as the importance of pilot schemes and “quick wins.” Progress has been patchy, and there is plenty more to be done. However, enough work has been surveyed to imply strongly that the technical issues are not intractable, even if managerial and political problems require a great deal of will and skill to solve. And as will be seen in the next section it is possible to launch national initiatives where the costs are dramatically smaller than those associated with traditional large-scale IT projects, in part because the main effort is in persuading the government to publish data in SW formats that support rapid reuse and exploitation outside government.

20.2 Examples: The Release of Public Linked Data in the UK

The challenges outlined in the previous section are pervasive throughout the public sector, and SW approaches must be alive to the dangers of ignoring them, especially as in [Sect. 20.1.3.8](#) because inaccurate negative perceptions of the SW can be extremely damaging at the crucial early stages of a project. In this section, two example systems will be examined that look at two aspects of the same problem, the provision of public

■ **Table 20.1**

Some common misconceptions about the Semantic Web (From [74])

Misconception	Reality
Everyone must agree to the same terminology to enable data and information sharing.	Different terminologies can be used by different departments, and linked to each other to ease sharing and communication.
Ontologies are typically large and complex.	Heavyweight and complex ontologies encode domain knowledge. Such ontologies are not always needed. Much can be done using relatively lightweight ontologies.
Ontologies are expensive to design, build, and maintain.	Some ontologies encode a great deal of domain knowledge and can be expensive to build. In these heavyweight ontologies the larger the potential user community the more the cost of construction is offset. Lightweight ontologies can have wide applicability and can be very cost-effective to build in terms of overall utility to the community [75].
Information and data must be taken out of current knowledge management practices, expensively converted to RDF, and then everything must be replaced with new standards and technology.	RDF creation can be automated, using simple scripts, APIs, or conversion languages (e.g., GRDDL). Data and information can be kept in their current formats, and cached or exported in RDF.
Providing access to data and information will benefit consumers and competitors, but there are no quick wins for the provider.	In the long run, exposing data and information will provide gains for the owner as well as for the whole network, just as exposing documents provided gains when the WWW took off. In the short term, much reuse of information is facilitated, which results in quick wins for any organization with a large quantity of distributed legacy data in heterogeneous formats.
The promiscuous release of data and information will be a privacy nightmare.	Standards are being developed to control access and reuse policies. In the meantime, as with conventional databases and Web technologies, organizations can pick and choose what data and information to expose and share.

datasets for public use. Specifically, they are both addressing the challenge as in [Sect. 20.1.3.3](#) of publishing information, yet it is clear to see that approaches to this challenge in this space will have ramifications to many if not all of the other challenges. *Publication* demands a stance on *representation* (an obvious question: should data be re-represented?). A sufficiently well-crafted representation, combined with the publication of data and the ability to link with other databases and knowledge bases, will allow

bottom-up *integration* to be performed by people with specific information needs, whether they are members of the government, engineers of *services*, or simply citizens writing or using mash-ups to re-present information, empowering them in their local communities, or helping them hold their government to account. Conversely, a poor representation will render integration hard if not impossible. The facilitation of *search and retrieval* is of course the point of publication. Protecting *privacy* is clearly vital, and needs to be prominently addressed in any publication strategy. *Reengineering* will be required, and furthermore negative *perceptions* will have to be addressed to avoid government inertia.

The two examples discussed below have been selected to illustrate in particular two very specific points. The first example of AKTive PSI will demonstrate how the process of re-presenting information can be conducted in order to defuse negative perceptions of reengineering, change management, and the SW itself; its focus is therefore primarily methodological. The second example, of data.gov.uk, focuses on the technical means for supporting representation and publication for the citizen to allow integration and provision of new inferred information, mash-ups, and services from existing information.

20.2.1 AKTive PSI

The first example is a detailed consideration of an early pilot or proof of concept for the integration of government information using reusable and linkable formats suitable for SW technology, the AKTive PSI project [39], which informed government thinking on information policy in the UK to an unusual degree in the SW world. The small-scale success of AKTive PSI in 2006–2008 paved the way for the more ambitious data.gov.uk site, which followed in 2009–2010. The ideas behind AKTive PSI were an important step in the “semanticization” of eGovernment, and understanding of how to represent government information to promote reuse in accordance with the Web’s and the SW’s model of value. The focus in this section is on the pragmatic reengineering of the systems and processes that use information, and how mandating good information representation, annotation, and publication policies is extremely important. Where possible, complexity of modeling is sacrificed to low-overhead practices, resulting in a tractable process even for small sectors of local government [74].

20.2.1.1 Context

The Office of Public Sector Information (OPSI) is responsible for the management of all of the UK government’s intellectual property, including setting standards, delivering access, and encouraging the reuse of PSI. OPSI also has an important role as a regulator of holders of public-sector information (e.g., the Met Office, Ordnance Survey) for their information-trading activities.

Information policy developed rapidly in the UK in the early twenty-first century, with Freedom of Information legislation as well as adoption of EU Directives, but for a long time

no large-scale work was carried out to research the potential for reuse using SW technologies and approaches. OPSI initiated a small project together with a UK-based project Advanced Knowledge Technologies (AKT – <http://www.aktors.org>), called AKTive PSI [39], as an exemplar to show what could be achieved if public-sector information was made available for reuse in an enabling way [40, 76]. Throughout the project, there were regular consultations with many governmental organizations, including the London Boroughs of Camden and Lewisham (local government), Ordnance Survey (the UK national mapping agency), The Stationary Office (the privatized but official publisher of the UK government), The Met Office (the UK's national weather service), The Environment Agency (the body with responsibility to protect the environment), The Office for National Statistics (ONS, the UK's provider and publisher of official statistics), and several others.

Some of the direct outcomes of AKTive PSI were: (a) the *London Gazette* (the official newspaper of public record in the UK – <http://www.london-gazette.co.uk/>) building OWL ontologies to represent parts of their data, and working toward publishing these data in RDF; (b) the development of a URI schema used to generate URIs for government official legislations and copyright statements; (c) Camden Borough Council added a SW engineer to their staff force to help the council in their effort to join the Web of linked data; and (d) the Ordnance Survey continuing their work and research in SW, building a number of ontologies and releasing several datasets.

The initial aims of the project were to draw together a sufficiently large set of heterogeneous information from a selection of public-sector organizations in order to: (a) explore how SW technology could help turn government information into reusable knowledge to support eGovernment; (b) investigate the best practical approaches to achieve this goal, in terms of collecting data and constructing ontologies; (c) show how data can be integrated, and identify existing government taxonomies that are useful for this task; and (d) provide evidence of the added value from undergoing this process. Note the strong pragmatic bias in these goals.

To help focus the requests for content, information was collected from the geographical area covered by two of the participating London local authorities, Camden and Lewisham.

20.2.1.2 Public-Sector Datasets

Several organizations who participated in AKTive PSI made some of their databases available for the project (🔗 [Table 20.2](#)). Note the heterogeneous nature of these data, and the standard formats in use. A number of scripts were developed to convert them to RDF automatically, in correspondence with their designated ontologies.

20.2.1.3 Ontology Construction

One of the AKTive PSI principles for building SW applications was to ensure the ontologies built for the provided datasets were of low complexity and limited in scope and size. Small ontologies are cheaper and easier to build, maintain, understand, use, and

■ **Table 20.2**

Datasets provided to AKTive PSI, the number of RDF triples generated for each dataset, and a description of what the data describe (From [39])

Camden Borough Council			
Land and property gazetteer	2.3 M	Excel	Properties in Camden, full address, coordinates, type (residential/nonresidential/mixed).
Food premises	84 K	Excel	Food-related premises in Camden, their business names, hygiene inspection results, addresses, (e.g., restaurant, school, bar).
Local businesses	170 K	Excel	Businesses in Camden, names, addresses, contact info, and type of business.
Licenses	100 K	MSSQL	Licenses for businesses in Camden, their addresses, license types, and expiry dates.
Councillors and committees	29 K	Excel	Councillors and committees, subcommittees, who sits on which committee, councilor's personal information.
Meeting minutes	106 K	Text	Web pages of committee meeting's minutes.
Lewisham Borough Council			
Land and property gazetteer	4 M	Excel	Properties in Lewisham, their full addresses, and coordinates.
Property tax bands	10 K	Excel	Tax property references, description, rate payers, rate value, and one-string addresses.
Ordnance Survey (data for Camden and Lewisham only)			
Address layer 1	768 K	XML	Data about buildings, addresses, and coordinates.
Address layer 2	11.7 M	XML	Data about buildings, addresses, and coordinates, and building classifications (e.g., hospital, university).
PointX POI	467 K	XML	Various landmarks and businesses, with names, addresses, and coordinates.
The Stationery Office London Gazette (entire database was provided, but only that listed below was used)			
Administration notices	120 K	Text	Notices for the appointment of administrator for corporate insolvencies.
Deceased estates	3.2 M	Text	Decease notices of individuals, names, addresses, description, and date of death, address of representatives.

commit to. It transpired, as is often the case, that no databases held by the participating organizations required more than a relatively small number of concepts and relationships to represent the stored information.

When building these applications it was important to show that ontologies are not hard to build if their purpose is representing databases and information assets of circumscribed scope, and that it is not necessary for everyone to come to a common,

agreed consensus on vocabulary, but that through ontology mapping techniques, local terminologies can also prove very useful. It is interesting to note that the average number of classes in the ontologies was 30, with a median of 10.

► *Figure 20.1* shows an example of an ontology from AKTive PSI that describes, in very simple terms, Camden's Land and Property Gazetteer. Each council in the UK has a database for the properties that exist in their administrative region, with simple data such as address, property type, ID, etc. The ontology in ► *Fig. 20.1* was built manually to model the concepts necessary to represent these data. Manually building ontologies for databases of such limited scope proved to be practical and cost-effective, especially as this ontology was reused for other similar databases held by other councils.

Representatives from the councils of Camden and Lewisham were given a 1-day course on ontologies, which covered ontology editing tools, and best practices and methodologies for ontology design and construction. The course was aimed at giving them the necessary basic skills and knowledge to start creating and exploring with ontologies. These measures of hand crafted ontology building and small training courses are relatively low overhead, thereby helping meet the challenges of representing and integrating information and reengineering and change management, while simultaneously ensuring that negative perceptions are minimized.

20.2.1.4 Generating RDF

The knowledge representation language of choice for the SW is RDF, which supports linking via the use of URIs, and reuse across data stores. Representing new data in RDF is one thing, but of course much, probably most, government data will be in legacy formats, raising the important issue of re-representation in RDF without putting too much of an administrative overhead on the process. There are several ways in which this can be done. For example, it is often the case that the data are maintained in live relational databases as part of the company's information flow and data network. In such cases it becomes necessary to use technologies such as D2RQ (<http://www4.wiwiw.fu-berlin.de/bizer/D2RQ/>) and Virtuoso (<http://www.openlinksw.com/dataspace/dav/wiki/Main/VOSRDF>), which allow for the data in RDBMS to be accessed in RDF.

The participating organizations provided AKTive PSI with data dumps extracted from their databases. The purpose was not to run live services off their networks, but to showcase SW technology as proof of concept. Therefore, in this project the aim was to transform the data into RDF and store it in triple-stores.

From an ontology it was possible to create instances by running simple scripts over the data/information to produce RDF. The scripts were hand-rolled specifically for the database and ontology which they were linking (reused across similar databases and ontologies). Although they were manually built, a framework for semiautomatic script generation was clearly conceivable. Most of the scripts were written using the Jena API, and were thus reusable and easy to tune for new datasets and ontologies. The participants were shown the relative ease of converting legacy data to RDF using free and simple technology.

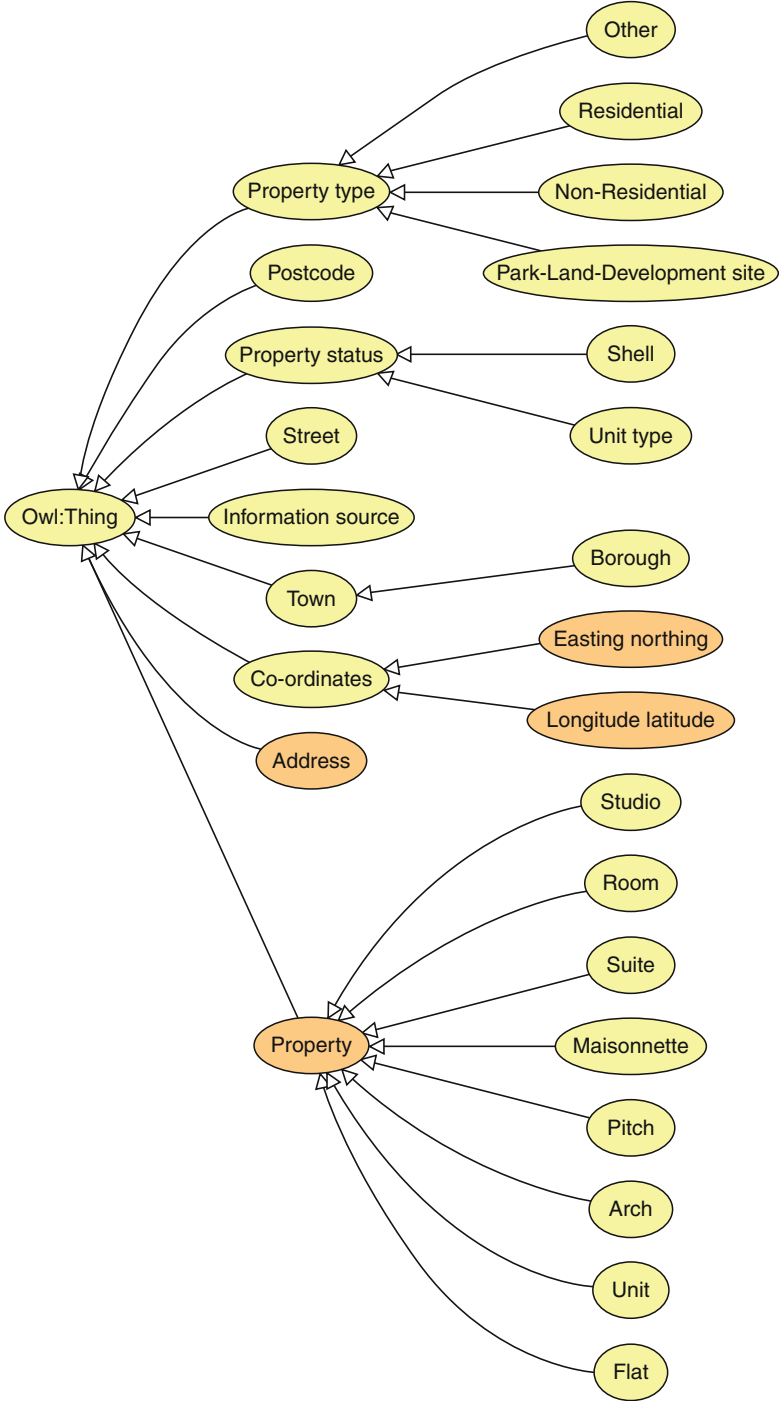


Fig. 20.1

Ontology for the Camden Land and Property Gazetteer, eGovernment

Although small ontologies were needed to represent the data, a scalable KB to hold the millions of RDF triples generated was also required. AKTive PSI used the 3Store (<http://www.aktors.org/technologies/3store/>), an RDF triple-store developed in the AKT project, to store the generated RDF files. This triple-store provides a SPARQL end point, a servlet that accepts SPARQL queries and returns results in XML. Publishing RDF in accordance with best practices [77] rendered the data viewable with general-purpose RDF browsers (e.g., Tabulator (<http://www.w3.org/2005/ajar/tab>) [78]). A key principle is that all entities of interest, such as information resources, real-world objects, and vocabulary terms should be identified by URI references. Once these are in place one can insist that they should be de-referenceable, meaning that an application can look up a URI over the HTTP protocol and retrieve RDF data about the identified resource.

20.2.1.5 Migrating to the Web of Data

Reuse of URIs increases connectivity between the published data and thus facilitates discovery of related data, addressing the search and discovery challenge discussed above [79]. Ontologies support integration using “soft” mappings between concepts and instances that queries or data browsers can follow to find similar or duplicated entities. AKTive PSI used the *owl:sameAs* property to link any mapped entities. By connecting KBs in this way much greater flexibility and querying power was available than the original data structures could provide.

This measure helped demonstrate the added value of using SW technology for publishing and using data. Forming a bigger semantic network by integrating the KBs containing all the data is clearly important in this context, easing communication and data/information exchange between the partners.

Two levels of mappings were performed.

- Mapping of local ontologies. Automatic ontology mapping has been the focus of much research for a number of years [80], and many tools have been developed for this purpose. However, to ensure accuracy, human supervision and intervention is still a necessity when mapping ontologies. Because of the relatively small size of the ontologies, this was not a difficult task in AKTive PSI. In fact, it was easier to map them manually than to correct automated mappings – an important by-product of the effort to address negative perceptions of the costs of implementing SW technology. Because of their expertise in the domain, the individual organizations provided important input to this process; this again was made possible because of the relatively simple level of the ontologies in question.

As will be shown later, mapping does not have to be complete to be useful. Much value can be drawn from mapping even a small number of concepts.

- Mapping of instances. Because of the data-centric approach it was important to map the instance data to each other as well. Instance mappings have to be done automatically as even in simple domains there will be a lot of instances to map. Automation is

done with simple scripts that search for duplicates of specific types of instances (e.g., postcodes, airplane models). An *owl:sameAs* link can be automatically added between the corresponding instances once such a mapping is found.

These processes create several files that contain RDF *owl:sameAs* triples linking various parts of the data. These files are stored separately from the data, and invoked when querying. To retrieve data from the knowledge base, the applications use SPARQL queries. Because the ontologies and data have been linked as described, it is possible to extract information from multiple data sources.

20.2.1.6 Mappings

Two ontologies for datasets from Lewisham Borough Council were developed, each with classes representing *property*, *address*, and *postcode* (i.e., equivalent to US zip codes). These concepts were linked with *owl:sameAs* to indicate that they represented the same concepts. There were also many simple mappings, such as between the concept *Premises* from the Food Premises ontology of Camden to the *Property* class in the Land and Property ontology. The CROSI mapping library (<http://www.aktors.org/crosi/>) was used for automatically generating these mappings.

But even simple mappings can be powerful, such as between instances of postcodes, for example, the instance *postcode_N6_6DS* in one KB maps to the instance *pc_N66DS* in another. Since these instances really do refer to the same object it is possible to infer far more by noting the identity. In fact, simply linking to one data object (the postcode) was generally enough to glean useful information from various datasets for the creation of interesting mash-up applications.

20.2.1.7 Mashing up Distributed KBs

Once data and information are available in easily parsable and understandable formats such as RDF, mash-ups become much easier to generate by searching RDF KBs and mashing up data on-the-fly, one of the advantages of linked data. Two such mash-ups were created in AKTive PSI. The aim of building these mash-ups was to demonstrate the benefit of exposing these data to the consumer, and the relative ease with which they can be constructed from semantically represented knowledge.

The Camden Food Premises database gives information about the hygiene check results and health risk of various premises around the Camden area that handle food. The risk categories are given a level between A, which is high risk, to E that is low risk, and is based on the cleanliness of the premises, compliance with regulations, type of preparation that is performed, etc. The Food Premises database contains lots of information on these properties, but displaying this information on a map is difficult because the geographical coordinates are missing from this particular dataset.

However, the Ordnance Survey's Address Layer and Points of Interest (PointX) datasets contain easting and northing coordinates for businesses and properties. The instance mapping of postcodes performed earlier helped to cut down the search space for finding matching addresses in the datasets. Indeed, once matches had been found it was possible to assert them as being the same, thereby avoiding the need for searching again.

To create the mash-up, a number of SPARQL queries were written that searched for each premise's address from the Food Premises dataset in each of the two OS datasets and once a match is found the coordinates are retrieved and the premise displayed on a Google map. The information from Food Premises together with the mapping between the datasets provides extra context to instances from either dataset. The PointX dataset gains access to the risk level of the food premises (as well as the implicit knowledge that the premises are used for preparing food), and the food premises dataset garners exact coordinates for the premises. [▶ Figure 20.2](#) shows a simple Google Maps mash-up that uses the mapping to provide a visual display of the food premises dataset.

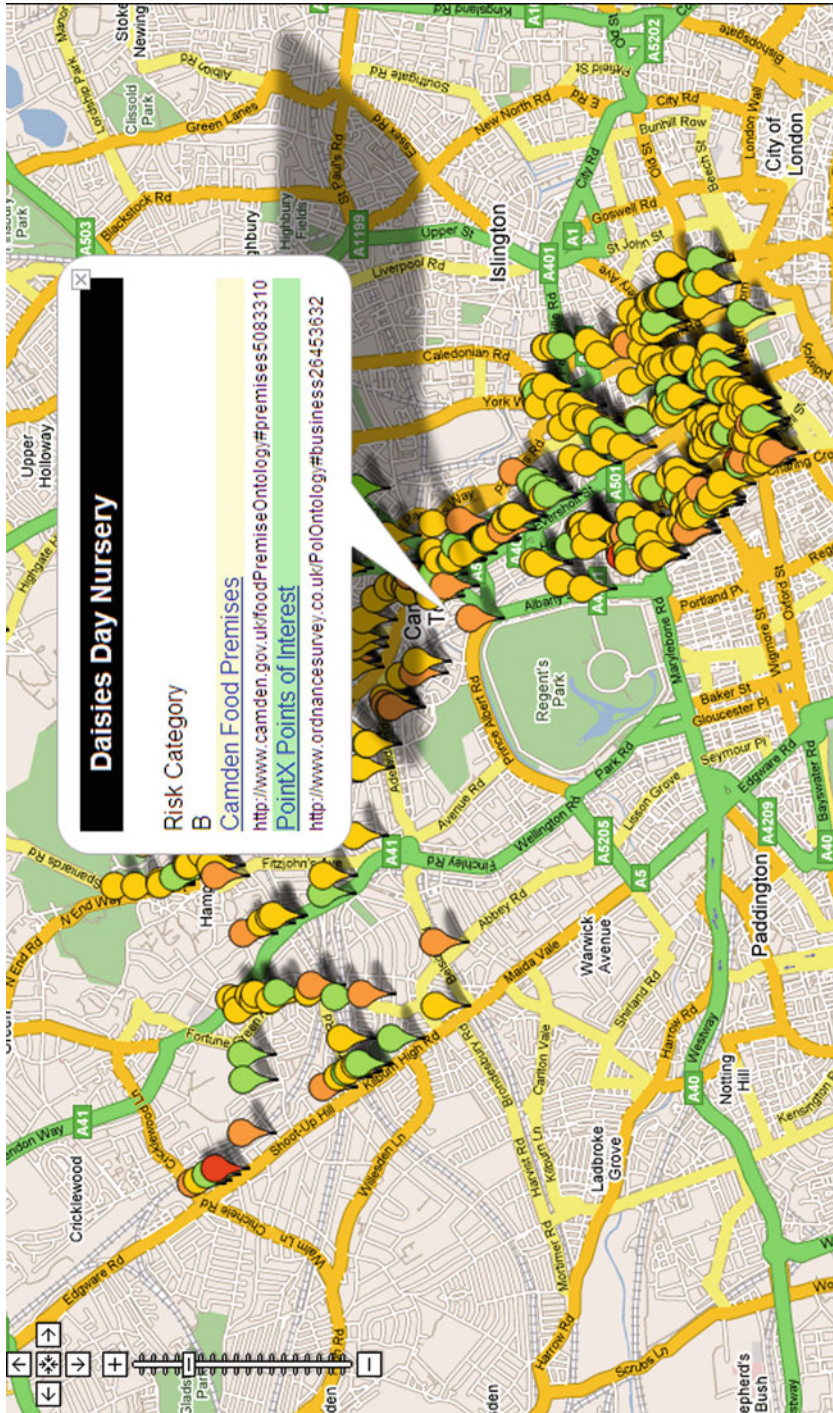
This type of mash-up promotes public awareness and indeed commercial competition. For example, it became evident that one particular business that scored within the high-risk category has glowing customer reviews on restaurant review sites across the Internet.

20.2.1.8 Inconsistencies

Data and information integration from multiple sources adds the value of knowledge augmentation and verification. Integrating datasets can provide useful insights into the quality of the dataset for the data provider involved. For example, the Ordnance Survey's *Address Layer 2* dataset provides a list of businesses, including their addresses and their geo-locations, and similarly so does the PointX dataset. However, it was found that the two lists of businesses do not match, for instance some being present in one dataset but not the other. In some examples, the PointX dataset contained several businesses listed at the same address, while only one was listed in the OS Address Layer 2. Was this an error? Perhaps one business took over the building from another, but the lack of temporal information concealed the fact, or perhaps one business is sited in the same building on a different floor to another business. It is difficult to infer an answer, but the integration has at least provided some information about the quality of the datasets and made such comparisons and cross-matchings possible. As noted above with the more complex examples of OntoGov for instance, such models can promote reengineering by helping identify inconsistencies.

20.2.2 data.gov.uk

AKTive PSI was an important proof of concept that led the UK government thinking on data management and integration. As the concept of *linked data* became increasingly important as a stepping stone to the development of the fully machine-readable SW



■ Fig. 20.2
Google Maps mash-up of the Camden Food Premises dataset made possible by mapping the data to the OS Address Layer II and PointX datasets, eGovernment

[79, 81], allowing data from multiple and heterogeneous sources to be linked, integrated, and reused, the quantity of linked data available on the Web has grown dramatically.

Where AKTive PSI had demonstrated the viability of using SW technology to add value to government information by integrating it using RDE, ontologies, and SPARQL, a potential extension was to add government data to the linked data Web by the use of resolvable URIs. If resolvable URIs could be used for reference in public-sector data, then it could be added to the Web of linked data with all the benefits that entail [82]. This is the approach underlying data.gov.uk.

20.2.2.1 Introduction and Context: First Steps

The early history of data.gov.uk is outlined in [83]. In June 2009, the then UK Prime Minister asked Tim Berners-Lee and Nigel Shadbolt to act as Government Information Advisors. Their terms of reference included:

1. Overseeing the creation of a single online point of access and working with departments to make this part of their routine operations
2. Helping select and implement common standards for the release of public data
3. Developing Crown Copyright and “Crown Commons” licenses and extending these to the wider public sector
4. Working with the Government to engage with the leading experts internationally working on public data and standards

The project was made public with an online call for help from the UK Cabinet Office that established one of the key principles of data.gov.uk, citizen participation: “From today we are inviting developers to show government how to get the future public data site right – how to find and use public sector information” [84]. By setting up a Google Group the Cabinet Office started to collect, and still collects, Web-user opinions about the data.gov.uk site (<http://groups.google.com/group/uk-government-data-developers>). In this online e-mail group there were more than 2,400 members (as of August 2010) to participate in daily message exchanges about all sorts of topics related to Open Public Sector Information (PSI) and Web technologies. During the first phase of data.gov.uk access to the portal was restricted only to members of the Google Group. The goal was to use this online group as input to improve the future site. During the following 3 months opinions and questions were gathered and the group listed valuable issues – technical, social, organizational, cultural, etc. – that needed to be understood and confronted in order to achieve the program’s ambitions to put government data in a position where they will be:

- Easy to find
- Easy to reuse
- Easy to license

The result of this collaborative process was made public in January 2010 when data.gov.uk was launched [85]. From that moment on, the portal allowed unrestricted

access and any Web user could make use of the data.gov.uk services and access more than 3,763 datasets.

Much of the work behind the scenes was around policy as much as technology. There were significant issues to be resolved to enable a permissive license for anyone to reuse data for any purpose – this has led to a new Crown Licence (<http://data.gov.uk/terms-and-conditions>). There was work to release significant amounts of Ordnance Survey UK Mapping data (<http://www.ordnancesurvey.co.uk/oswebsite/opendata/>) using the same permissive license and free of derived data constraints (i.e., the constraint that if one contributes data to another underlying data source – say a map – the owner of that underlying data has the rights to the contributed data). There was an early recognition that much important data lay in local government and not in the central government’s hands. This led to the establishment of a Local Public Data Panel (<http://data.gov.uk/blogs/local-data-panel>) charged with coordinating and promoting transparency and open data policies in local government bodies. And there has been a continuing recognition that this is all work in progress.

The data.gov.uk project’s first premise is to open data for reuse in any processable format – spreadsheets, database dumps and XML files, etc. The project has also a clear commitment to Web standards in general and Semantic Web technologies in particular (cf. [82]). Each dataset is to be transformed into linked data format after being released, a strategy corresponding to Berners-Lee’s call for “Raw. Data. Now!” [86] and his five-point scheme for publishing data [79]. Hence the data.gov.uk site combines Semantic Web and traditional Web technologies giving a single point of access to citizens for finding and reusing data through the following services:

- Semantic Web Features:
 - SPARQL end points organized by the government sector.
 - Linked datasets.
 - Practices and strategies for publishing linked data.
- Non-Semantic Web features
 - Forum and Wiki to continue promoting collaboration around the datasets.
 - Searching/browsing datasets. The site acts as a single point of access allowing users to quickly access data by areas of interest such as “crime,” “education,” “economy,” etc.
 - Ideas and applications repository, where users can submit and find applications that are already using PSI data.

One of the key efforts in data.gov.uk is to develop best practice and strategies for publishing PSI UK linked data. These practices are directed to any data publisher – part of the government or not – that wants to transform or create a PSI dataset into linked data format. Consistent with the earlier work on AKTive PSI, this process compares cost to benefit for adopting linked data as suitable technology for publishing PSI data. So far, the project has identified a number of practices as crucial [87], including URI design, versioning, provenance, and the development of core ontologies.

Note that the data.gov.uk approach (like the data.gov approach in the USA) addresses the challenge of integration in effect by outsourcing. By releasing data, and by supporting

the creative use of mash-up technology to bring data together in novel, informative, and surprising ways, these sites open up government data to the power of the Web's scale [82]. Many people can get together to find creative ways of amalgamating databases, and so the bottom-up integration is demand-driven, while actual government involvement need only be limited to the mandating of the representation or re-representation of data in formats designed to support linking. The rest of this section will discuss the formats in use.

20.2.2.2 Government URI Structure

One of the first documents released by the Cabinet Office as part of the data.gov.uk effort was *Designing URI Sets for the UK Public Sector* [88]. This document describes the structure of a Government URI and, together with the Cool URI definition from W3C [89], represents a guideline for minting URIs based on established and emerging good practices in the linked data community. They also meet specific needs for the UK public sector. In summary, these practices include:

1. Use of data.gov.uk as the domain to root those URI sets that are promoted for reuse
2. Organization of URI sets into “sectors” (e.g., education, transport, health) with a lead department or agency
3. Consistent use of metadata to describe the quality characteristics of each URI set

► [Table 20.3](#) shows a summary of the URI structures with examples for various types of URI.

■ **Table 20.3**

UK government URI structures and examples [88]

URI Type	URI Structure	Examples
Identifier	http://{domain}/ id /{concept}/ {reference} or http://{domain}/ {concept}/{reference}#id	http://education.data.gov.uk/id/school/78 http://education.data.gov.uk/school/78#id http://transport.data.gov.uk/id/road/M5/ junction/24
Document	http://{domain}/ doc /{concept}/ {reference}	http://education.data.gov.uk/doc/school/ 78
Representation	http://{domain}/ doc /{concept}/ {reference}/{doc.file-extension}	http://education.data.gov.uk/doc/school/ 78/doc.rdf
Definition of the scheme concept	http://{domain}/ def /{concept}	http://education.data.gov.uk/def/school
List of scheme identifiers	http://{domain}/ doc /{concept}	http://education.data.gov.uk/doc/school
Set	http://{domain}/ set /{concept}	http://education.data.gov.uk/set/school

As can be seen, the domain of the data is indicated by the first fragment of the URI to be minted. The URI scheme defines entities for both the instance and the schema level establishing a clear separation for individuals and definitions by using the “def” and “id” nomenclature. To differentiate documents from information resources it introduces the “doc” suffix. For easy dereferencing of lists of instances – for instance lists of schools – the document defines the URI “set” structure. When resolving a URI “set” one expects to retrieve a collection of the entities represented by that set.

20.2.2.3 Versioning

Legislation, geographical boundaries, and local authorities are just some examples of PSI datasets that change over time. Statistical datasets contain an implicit versioning mechanism, because time is normally one of the axes in multidimensional datasets. This means that the time dimension is treated as a series and the statistical observation has validity only over a temporal instance. For example, one does not tend to create versions of the number of tons of CO₂ emitted in the South West of England for different years; instead that knowledge is represented with a multidimensional dataset where geography and time are dimensions that give contextual meaning to the observation (ontologies to describe such statistical information are given in the “ontologies” subsection below).

Even though most of the PSI data are statistics, there are also important nonstatistical datasets where it is very important to relate to pieces of information that were valid in the past. For instance, it is important to track shifting scheme classifications, like the UK Standard Industrial Classification of Economic Activities (SIC) that released different versions in 1992, 2003, and 2007. The approach to implement such type of versioning in linked data is to use Named Graphs [90]. A graph asserts a set of statements and the graph gets annotated with the temporal validity of the information in it. The ontologies used to annotate the validity of a graph and the relations between different versions of the same resources are mainly FOAF [36] and Dublin Core [91].

► *Figure 20.3* shows an example of a linked data resource, a UK school, which has changed its name. The end of validity of the older graph is stated by the assertion of the Dublin Core predicate *isReplacedBy*. This solution also asserts metadata about the version in each of the graphs so that a software agent that visits one of these versioned graphs will be aware of the temporal validity of the information and how to navigate to other versions of the same data.

20.2.2.4 Provenance

Alongside with versioning it is important to provide information about the provenance of the data. Provenance is not just about the source of the information but also about the ways in which the data have been manipulated in the process of publishing them. The approach adopted is the same as in versioning, where named graphs play a key role in



■ Fig. 20.3

Example of versioning, recording the change of name of a school, eGovernment

stating metadata about a dataset. The W3C Provenance Incubator group (http://www.w3.org/2005/Incubator/prov/wiki/W3C_Provenance_Incubator_Group_Wiki) is investigating the state of the art and developing a roadmap in the area of provenance and Semantic Web Technologies. The Open Provenance Model (OPM) [92] is a model that has been embraced by both the W3C Provenance Group and the data.gov.uk project as a standard to represent the provenance of the data. This model among other things consists of an RDF vocabulary, the Open Provenance Model Vocabulary (OPMV), which provides terms to enable practitioners of data publishing to publish their data responsibly.

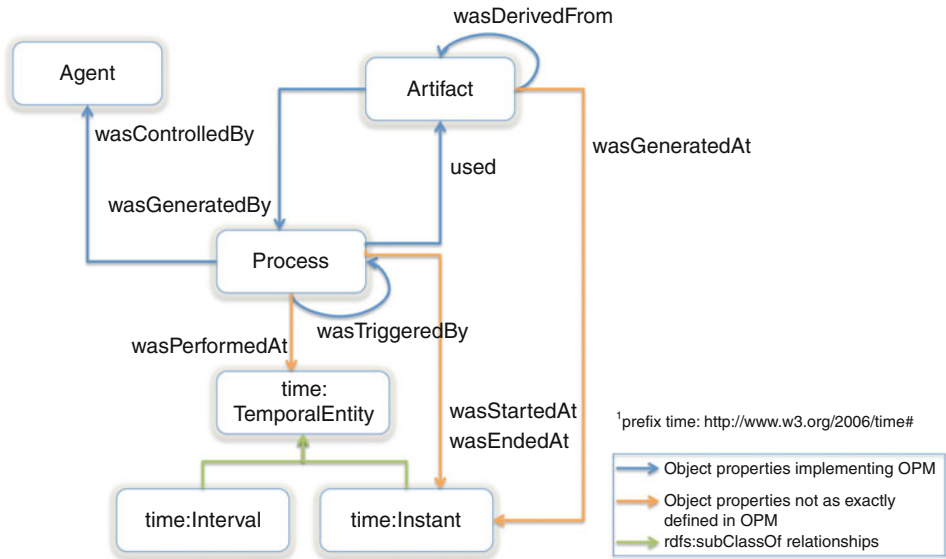
The Core OPMV predicates are represented in ▶ Fig. 20.4. The main entities that make up this model are *agents*, *artifacts*, and *processes*. As in almost any other workflow model agents represent the actors that trigger and control the processes; Processes refer to any action performed over artifacts; and artifacts are the input and outputs of the processes.

In the data.gov.uk project this model has been adopted to represent all the different actions that take place in the process of publishing linked data.

20.2.2.5 Core data.gov.uk Ontologies

Another activity in the data.gov.uk project is the development of core ontologies, which act as references for publishing linked data for the UK government. Two efforts within this activity have attracted attention from the linked data community: the SCOVO and SDMX schemas for describing statistical data, and an ontology for describing organizational structures.

SCOVO/SDMX: Statistics are probably the most common type of information in PSI. Almost every PSI dataset contains a multidimensional data structure with temporal series as one of the axes, from traffic flows and CO₂ emissions to indices of deprivation and government expenses. SCOVO (<http://sw.joanneum.at/scovo/schema.html>) was the first ontology describing multidimensional structures that was explicitly designed to describe statistics. First versions of the statistical data in data.gov.uk used this ontology as main schema.



■ Fig. 20.4

Core definitions of the Open Provenance Model Vocabulary (<http://open-biomed.sourceforge.net/opmv/ns.html>), eGovernment

Unfortunately the semantics in SCOVO are not powerful enough so as to describe standards like the Statistical Data and Metadata eXchange standard (SDMX – <http://sdmx.org/>), supported by numerous institutions and by the UK Office for National Statistics (www.statistics.gov.uk/). As part of a consultation process started by data.gov.uk, a different project was launched to bring together SDMX and SCOVO. This seeks to provide a forum to agree on an SDMX representation for the RDF information model. The result of that effort is a vocabulary, which, by using RDFS and OWL, extends SCOVO to represent statistical observations. Moreover, this vocabulary uses SKOS (<http://www.w3.org/2004/02/skos/>) to represent classification schemes and code lists (see ► Fig. 20.5).

The main differences between using SDMX/SCOVO and using just SCOVO are the representation of code lists (SKOS), time series, and data groups within a dataset [93]. SCOVO and SDMX are compatible and in fact SDMX extends SCOVO. It is important to notice that both ontologies will coexist in the linked data cloud. SCOVO is a lighter ontology than SDMX, easier to understand and more suitable for simple statistics. On the other hand, SDMX works better for complex cases and for datasets already in the XML representation of SDMX.

Organizational Structures Ontology: data.gov.uk has been also immersed in the definition of an organizational ontology. A survey on previous ontologies to describe organizational structures (<http://www.epimorphics.com/web/wiki/organization-ontology-survey>) discovered that most of them were designed to fit their own purposes and that generalized definitions of organizations are minimal. The organizational ontology (<http://www.epimorphics.com/public/vocabulary/org.html>) that the data.gov.uk group, led by

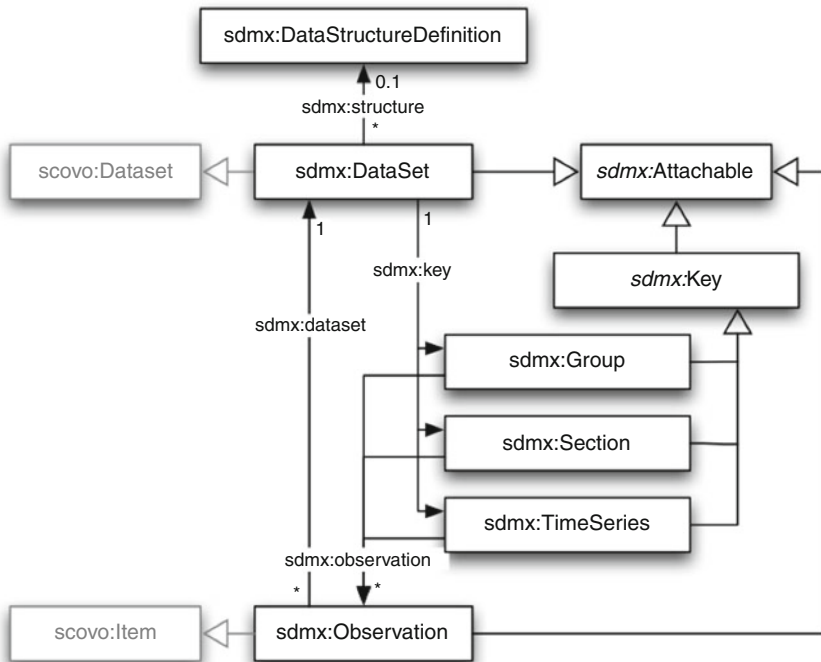


Fig. 20.5

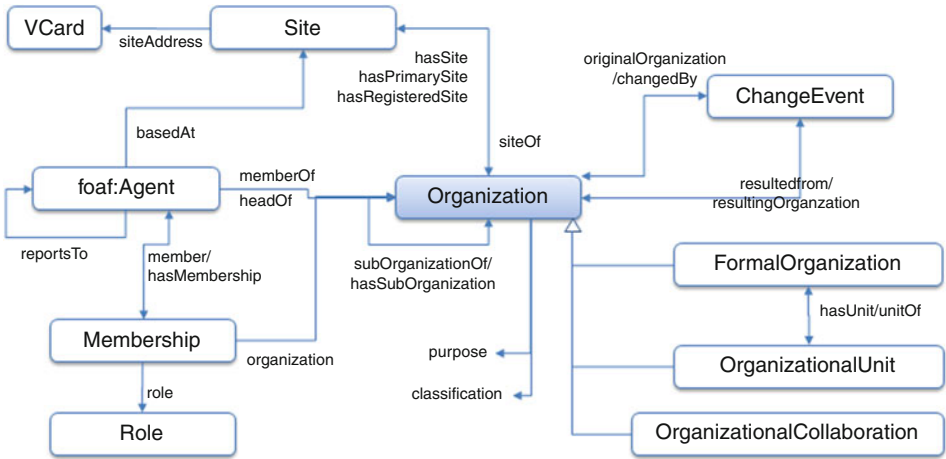
Part of the SDMX RDF Information Model (From [93]), eGovernment

Epimorphics, developed was critiqued by the Linking Open Data (LOD) community and the schema went through several iterative drafts (<http://esw.w3.org/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>). The result is a simple ontology suitable to represent government bodies such as departments, ministries, and secretaries, and the relationships between them.

The ontology reuses other vocabularies such as FOAF, Dublin Core, and OPMV. The core of the ontology is represented by *organizations*, *organizational units*, and *agents* (FOAF). It enables the representation of more complex structures by specialization of these concepts. The other important part of the ontology is the representation of relationships between agents and organizations: who reports to whom, and which units are part of others. For this, the ontology has predicates like *memberOf*, *hasMember*, *reportsTo*, *hasMembership*, etc. The ontology, in essence, is simple (see Fig. 20.6) but it can be extended to fit complex structural organizations. In fact, this ontology can be reused for representing not just government organizations but also other types of structured entities.

Other components of this ontology provide schema definitions for:

- *Locations*: classes to represent *sites* and *addresses* together with other predicates (*baseAt*, *hasSite*, *siteOf*, etc.) to link these with organizations, agents, etc.
- *Projects and Other Activities*: the class *organizational collaboration* (a subclass of organization) represents a collaborative action between one or more organizations.



■ Fig. 20.6

Organizational ontology overview, eGovernment

20.2.2.6 Linked Datasets

The imperative for any Open Data Government initiative is to have the data available on the Web, ideally in machine-readable form but even better in linked data format. One needs to recognize that for many releasing data is a journey and that the “best should not be the enemy of the good.” This is one of the reasons behind Berners-Lee’s star system for describing Government datasets (<http://www.youtube.com/watch?v=ga1aSJXCFe0>) – the rating system simply stated is:

- Make your stuff available on the Web (whatever format).
- Make it available as structured data (e.g., Excel instead of image scan of a table).
- Use an open standard, not a proprietary format (e.g., CSV instead of Excel).
- Use URLs to identify things, so that people can point at your stuff.
- Link your data to other people’s data to provide context.

To this end data.gov.uk has begun to release datasets in linked data format. The ambition is that ultimately this would be the publication format of choice for all government data. In these datasets previous practices have been tested to prove their applicability focusing so far on two domains: *education* and *transport*. A third domain, *geography*, has been also put in place thanks to the collaboration of the Ordnance Survey, which has been an important actor in the UK PSI linked data participating in research and releasing key geographical datasets, and as noted earlier was prominent in AKTive PSI. Most PSI datasets contain a geographical dimension so it is extremely important to have access to an authoritative source in that domain. To date (2010) the main linked datasets under the data.gov.uk umbrella are:

1. *Education*: The education dataset is an RDF transformation of the UK Edubase database (<http://www.edubase.gov.uk/>). This dataset is available through an SPARQL end point and the resources in it are also exposed as linked data. It contains information about all the educational establishments across England and Wales, including the schools' type, whether they are religious, their numbers of pupils, and their geographical positions linked to the Ordnance Survey Administrative Geography.
2. *Ordnance Survey*: Ordnance Survey has released the Administrative Geography of Great Britain in linked data format. This is an important data hub for data integrators. In [94] this is demonstrated in the context of a case study where linked data from various PSI sources were integrated. The administrative geography from OS describes each of the types of administrative areas of the UK: European Regions, Unitary Authorities, Counties, Boroughs, Districts, and Parishes; and, more important, the spatial containments between them.
3. *Transport*: Two important transport databases are in the process of transformation into linked data: the public transport network database and traffic flows. The public transport data sources used are the National Public Transport Access Node (NaPTAN) and the National Public Transport Gazetteer (NPTG) databases. The former contains all the public transport stops from airports and ferries to buses and trains; and the latter is a topographic database of towns and settlements in the UK, providing a common frame of reference for NaPTAN.
4. *Multiple Indexes of Deprivation*: In collaboration with the JISC-funded Open PSI project (<http://www.openpsi.org/>), the Multiple Indexes of Deprivation database (<http://www.communities.gov.uk/documents/communities/>) has been transformed into linked data. This database contains rankings of different types of social, economic, and cultural indexes in the UK.
5. *Ministers*: The ministers' dataset contains a reference set of the current UK government, their ministers and secretaries as well as the relationships with the different UK government organizations. It uses the Organizational Structure ontology to represent such information in linked data format.
6. *NUTS Geography*: NUTS (Nomenclature of Territorial Units for Statistics) is a standard developed and regulated by the EU members and it is an instrument for reporting statistics. The data.gov.uk project transformed the latest version of the UK NUTS Geography into linked data format.

As a separate effort but in collaboration with data.gov.uk the EnAKTinG project has also released key datasets in linked data format (<http://www.enaktin.org/gallery/>). The EnAKTinG catalog of datasets contains information about population, CO₂ road emissions, energy consumption, mortality, Parliamentary data (MPs, Lords, and their recorded expenses), and crime offenses. The SPARQL end points of the data.gov.uk datasets (as of August 19, 2010) are given in [▶ Table 20.4](#).

In the same context, but not part of data.gov.uk, it is important to mention the openlylocal project (<http://openlylocal.com/>). This project is an effort to make UK local

■ **Table 20.4**

data.gov.uk accessible SPARQL end points

Dataset	SPARQL end point	De-referenceable URIs
Education	http://services.data.gov.uk/education/sparql	Yes
Ordnance survey	http://api.talis.com/stores/ordnance-survey/services/sparql	Yes
Transport	http://gov.tso.co.uk/transport/sparql	No
Multiple indexes of deprivation	No	Yes
Ministers	http://services.data.gov.uk/reference/sparql	No
NUTS geography	http://services.data.gov.uk/statistics/sparql	No

governments more transparent by accessing their information. As their home page declares they hold information about:

- 158 councils
- 9,946 councillors
- 5,793 committees
- 47,386 committee meetings
- 309 hyperlocal sites
- 29,205 documents
- 52,443 financial transactions

The data available from openlylocal.com can be retrieved in RDF format (see example in [Fig. 20.7](#)) and the site provides interesting links to other linked datasets like data.gov.uk, Ordnance Survey, and DBPedia. Despite the fact that UK is one of the leading countries in the PSI open data effort and that there is a clear commitment from the UK government to keep improving, the statistics from [openlylocal](http://openlylocal.com) (see [Fig. 20.8](#)) show that a very small portion of local authorities can claim to have published open data.

20.2.2.7 Linked Data API

The linked data API (<http://code.google.com/p/linked-data-api/>) gained much attention in the course of the data.gov.uk project. This open-source project tries to bring together Web developers and linked data technologies. Despite the simplicity of linked data technologies – RDF and de-referenceable URIs – some Web developers have raised issues regarding its adoption as mainstream technology. The linked data API provides tools to overcome this barrier by enabling access to the RDF data model through more developer-friendly technologies.

In the last years simple RESTful APIs have succeeded in getting adopted by Web developers, while key players on the Web like Yahoo!, Google, or Amazon offer access to

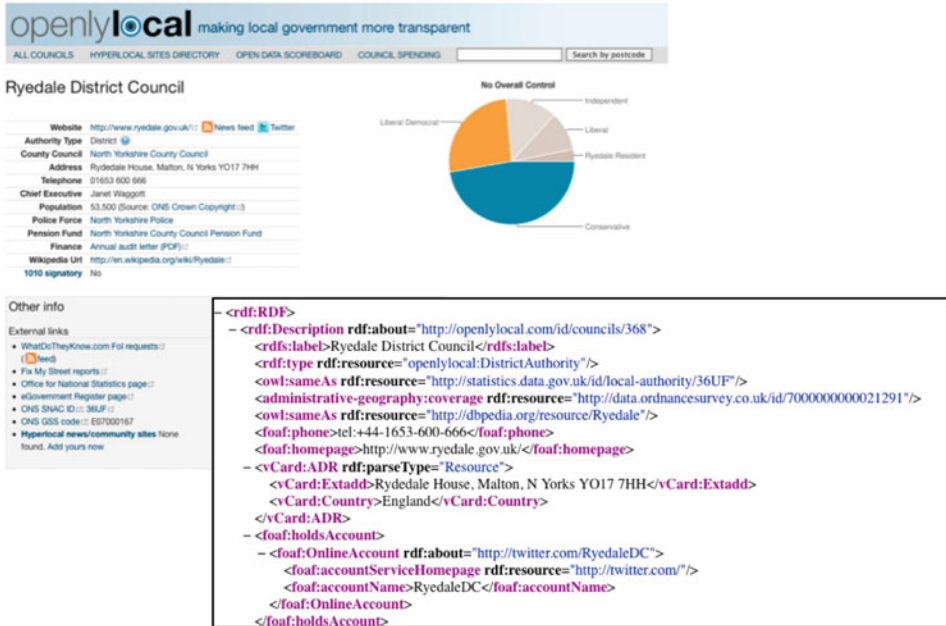


Fig. 20.7

openlylocal representation of Ryedale District Council showing HTML and RDF versions, eGovernment



Fig. 20.8

openlylocal Open Data Scoreboard as of August 2010 (<http://openlylocal.com/councils/open>), eGovernment

their data through them. The linked data API is a specification that describes how to expose the RDF model using RESTful services and, enables clients to process the data in various formats like JSON, XML, and RDF. The API is intended to be deployed as a proxy in front of a SPARQL query to support:

1. Generation of documents (information resources) for publishing linked data
2. Provision of sophisticated querying and data extraction features, without the need for end users to write SPARQL queries
3. Delivery of multiple output formats from these APIs, including a simple serialization of RDF in JSON syntax

The API maps URL patterns into SPARQL queries. The results of a SPARQL query are passed through two components, a Viewer and a Formatter, before giving a response to the data consumer. In essence the Viewer and the Formatter accommodate the answer in an adequate form. For instance, the following URL pattern exposes a search of schools by district name and different views for the result can be parametrized.

http://gov.tso.co.uk/education/api/school/district-name/{NAME}?_view = {view}

A Web developer can do a RESTful request to this service and by binding **{name}** and **{view}** variables he can use the request for his requirements.

► *Figure 20.9* shows part of the output for the given URL using “Vale Royal” as district name. The parameter **_view** is an API parameter that enables different output customizations for the same type of search – if this parameter is not given the default view is used. The linked data API provides pagination for search results. As can be seen at the top of ► *Fig. 20.9*, references to first and next pages of results are provided in the XML output. For each result of the search a link to the linked data resource is given. The result of the search keeps the original linked data URI so that the client application can always retrieve the RDF representation if needed.

The linked data API also provides parameters to customize the selection by changing the query. The requester of the service can modify the select, where, or sort clauses of the

```


<result format="linked-data-api" href="http://gov.tso.co.uk/education/api/school/district-name/Vale%20Royal?_view=short" version="0.1">
  <type href="http://purl.org/linked-data/api/vocab#Page"/>
  <isPartOf href="http://gov.tso.co.uk/education/api/school/district-name/Vale%20Royal">
    <type href="http://purl.org/linked-data/api/vocab#List"/>
    <hasPart href="http://gov.tso.co.uk/education/api/school/district-name/Vale%20Royal?_view=short&_page=1"/>
  </isPartOf>
  <first href="http://gov.tso.co.uk/education/api/school/district-name/Vale%20Royal?_view=short&_page=1"/>
  <next href="http://gov.tso.co.uk/education/api/school/district-name/Vale%20Royal?_view=short&_page=2"/>
  <itemsPerPage datatype="integer">10</itemsPerPage>
  <startIndex datatype="integer">0</startIndex>
  <items>
    <item href="http://education.data.gov.uk/id/school/111059">
      <label>Cuddington Primary School</label>
      <uniqueReferenceNumber datatype="int">111059</uniqueReferenceNumber>
      <establishmentNumber datatype="int">2196</establishmentNumber>
      <typeOfEstablishment href="http://education.data.gov.uk/def/school/TypeOfEstablishment_TERM_Community_School">
        <label>Community School</label>
      </typeOfEstablishment>
      <phaseOfEducation href="http://education.data.gov.uk/def/school/PhaseOfEducation_Primary">
        <label>Primary</label>
      </phaseOfEducation>
      <gender href="http://education.data.gov.uk/def/school/Gender_Mixed">
        <label>Mixed</label>
      </gender>
      <religiousCharacter href="http://education.data.gov.uk/def/school/ReligiousCharacter_Does_not_apply">
        <label>Does not apply</label>
      </religiousCharacter>
    </item>
    <item href="http://education.data.gov.uk/id/school/111056">
      <label>Wimboldsley Community Primary School</label>
      <uniqueReferenceNumber datatype="int">111056</uniqueReferenceNumber>
      <establishmentNumber datatype="int">2190</establishmentNumber>
      <typeOfEstablishment href="http://education.data.gov.uk/def/school/TypeOfEstablishment_TERM_Community_School">

```


■ **Fig. 20.9**

Example of Linked Data API response, eGovernment


query. For instance, by adding “`_sort = phaseOfEducation.label,religiousCharacter.label,`” one would sort the result set by the phase label – primary, secondary, etc. – and the religious character of the school.

As a complementary function to search, the linked data API also enables the retrieval of single database items like <http://gov.tso.co.uk/education/api/school/100869>, which represents a school. It implements content negotiation through HTTP content negotiation. Therefore, a request with “Accept: application/json” in the HTTP header would retrieve the JSON format. As an example, the following command with `curl` would give back the JSON object represented by  Fig. 20.10.

```
curl -L -H 'Accept: application/json' 'http://gov.tso.co.uk/education/api/school/100869'
```

In  Fig. 20.10, three different sections for the document are shown. The first contains the current data for the selected view. The other two provide links to other formats and views. By changing the `_view` parameter in the request a different representation of the data section could be obtained. For instance the following command produces a different view:

```
curl -L -H 'Accept: application/json' 'http://gov.tso.co.uk/education/api/school/100869?_view=location'
```

It has “location” as `_view` parameter, so the output includes the location of the school. As shown in  Fig. 20.11 the data section of this view includes the easting and northing coordinates and the detailed address of the school.

The linked data API is independent of programming languages or Web servers; at the time of writing (2010) two implementations – one in Java and another in PHP – coexist.

```
{
  "uniqueReferenceNumber": 100869,
  "about": "http://education.data.gov.uk/id/school/100869",
  "establishmentNumber": 6385,
  "gender": "http://education.data.gov.uk/def/school/Gender_Mixed",
  "label": "Southwark Small School",
  "religiousCharacter": "http://education.data.gov.uk/def/school/ReligiousCharacter_None",
  "typeOfEstablishment": "http://education.data.gov.uk/def/school/TypeOfEstablishment_TERM_Other_Independent_School"
},
{
  "format": "application/json",
  "hasPart": "http://gov.tso.co.uk/education/api/school/100869.json",
  "hasFormat": [
    "http://gov.tso.co.uk/education/api/school/100869.json",
    "http://gov.tso.co.uk/education/api/school/100869.rdf",
    "http://gov.tso.co.uk/education/api/school/100869.tt",
    "http://gov.tso.co.uk/education/api/school/100869.xml"
  ],
  "Item": "http://purl.org/linked-data/api/vocab#Item",
  "definition": "http://services.data.gov.uk/education/api#schoolsItem",
  "about": "http://gov.tso.co.uk/education/api/school/100869.json",
  "hasVersion": [
    "http://gov.tso.co.uk/education/api/school/100869.json?_view=admin",
    "http://gov.tso.co.uk/education/api/school/100869.json?_view=location",
    "http://gov.tso.co.uk/education/api/school/100869.json?_view=medium",
    "http://gov.tso.co.uk/education/api/school/100869.json?_view=performance",
    "http://gov.tso.co.uk/education/api/school/100869.json?_view=provision",
    "http://gov.tso.co.uk/education/api/school/100869.json?_view=short"
  ]
},
...
```

Data for Current View

Other Formats

Other Views

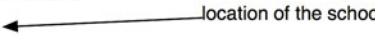
 Fig. 20.10

JSON representation of a UK school with the Linked Data API, eGovernment

```

{
  "administrativeWard": "http://statistics.data.gov.uk/id/local-authority-ward/00BEGH",
  "LSOA": "http://statistics.data.gov.uk/id/lsOA/E01003953",
  "about": "http://education.data.gov.uk/id/school/100869",
  "establishmentNumber": 6385,
  "LLSC": "http://statistics.data.gov.uk/id/llsc/GL140",
  "gender": "http://education.data.gov.uk/def/school/Gender_Mixed",
  "parliamentaryConstituency": "http://statistics.data.gov.uk/id/parliamentary-constituency/142",
  "urbanRural": "http://education.data.gov.uk/def/school/UrbanRural_Urban_10k_less_sparse",
  "districtAdministrative": "http://statistics.data.gov.uk/id/local-authority-district/00BE",
  "typeOfEstablishment": "http://education.data.gov.uk/def/school/TypeOfEstablishment_TERM_Other_Independent_School",
  "religiousCharacter": "http://education.data.gov.uk/def/school/ReligiousCharacter_None",
  "localAuthority": "http://statistics.data.gov.uk/id/local-education-authority/210",
  "uniqueReferenceNumber": 100869,
  "long": -0.07896999999999999,
  "eastng": 53352,
  "address": {
    "town": "London",
    "address1": "14 Derwent Grove",
    "address2": "East Dulwich",
    "postcode": "SE22 8EA"
  },
  "lat": 51.460700000000003,
  "label": "Southwark Small School",
  "MSOA": "http://statistics.data.gov.uk/id/msoa/E02000834",
  "censusAreaStatisticWard": "http://statistics.data.gov.uk/id/cas-ward/00BEGH",
  "northing": 175275
},

```



■ Fig. 20.11

Location view with the linked data API in JSON format, eGovernment

20.2.2.8 PSI Linked Data Mash-Ups and Applications

It is now possible to see Semantic Web mash-ups and applications where some of the practices described above have been applied, and where the datasets also played an important role, either by use of their linked data resources or via querying the SPARQL end points.

Most of the PSI linked data applications show how to integrate several linked data sources using a map as key part of their functionalities. The EnAKTing project studied how much linked data can be retrieved using UK postcodes as input [94], in an application integrating Parliamentary data, crime, hospital waiting times, and mortality rates. This use-case study brought up numerous issues that a Web or Semantic Web developer would need to face when developing this type of application. Other applications like “How Good is my area?” (<http://myarea.psi.enakting.org/>) or schools.openpsi.org have shown mash-ups around the Indexes of Multiple Deprivation dataset. The former ranks an area based on multiple rankings on a series of parameters and compares it with its surroundings. The latter is an interesting study that visually integrates school performances with the different sociocultural and economic aspects for a given location (see ▶ [Fig. 20.12](#)).

As part of the EnAKTing research, similar applications have been developed that consume linked data sources providing visualizations of integrated datasets. The “UK CO₂ Emissions Visualization” application (▶ [Fig. 20.13](#)) is a good example of such work and shows an application made with the integration of three linked data sources: Geonames, Ordnance Survey Administrative Geography, and CO₂ Emissions.

In the USA, the data.gov project has declared a clear commitment to Semantic Web technologies in part thanks to the promotion of these technologies from the Tetherless World Constellation at Rensselaer Polytechnic Institute (<http://rpi.edu/research/constellations/tetherlessworld.html>). In data.gov one can see the important role of these

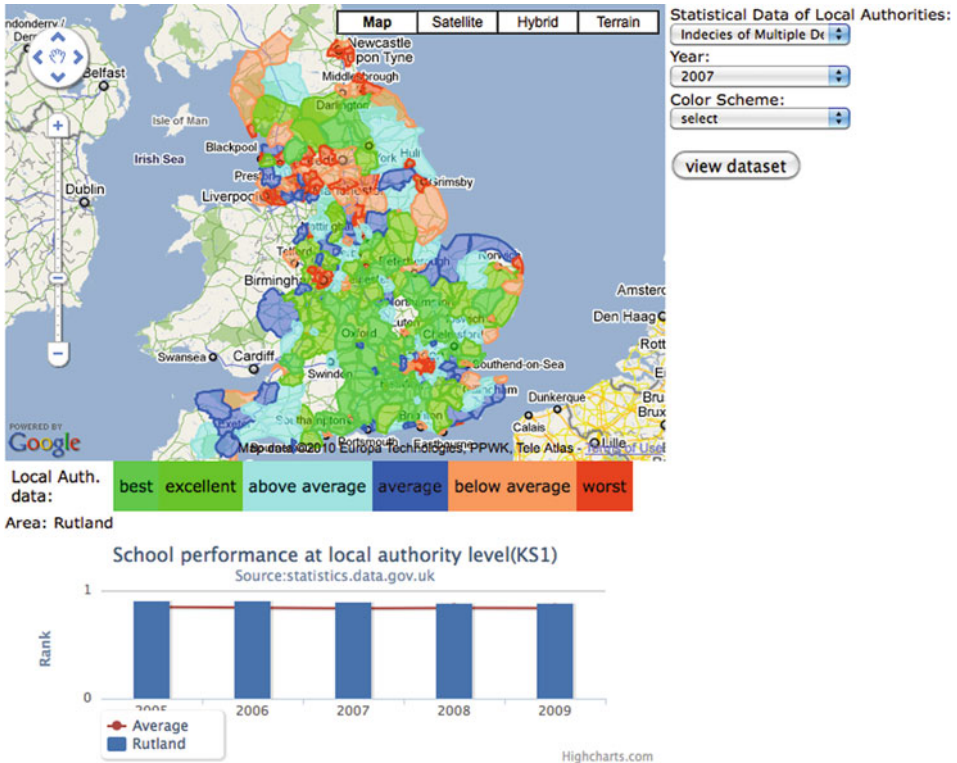


Fig. 20.12

OpenPSI school application, eGovernment

technologies with interesting mash-ups that show the use of linked datasets in a geographical context (see Fig. 20.14 – <http://www.data.gov/semantic/>).

20.3 Related Resources

Important resources for the student of eGovernment and the SW include:

- <http://www.recovery.gov/> is the address of Obama's website to monitor the stimulus. The US site that releases public-sector data is <http://www.data.gov/>
- The UK equivalent is <http://data.gov.uk/>
- The UK government's Public Sector Transparency Board can be found at <http://www.itetoreply.org/publicsectortransparencyboard/>. The UK transparency program is described in a letter from the UK Prime Minister David Cameron to Cabinet Ministers in 2010, available at <http://www.cabinetoffice.gov.uk/newsroom/statements/transparency/pm-letter.aspx>. A position paper written by OPSI reflecting its information policy and the influence of AKTive PSI is at <http://www.w3.org/2007/06/eGov-dc/papers/opsi-position-paper>

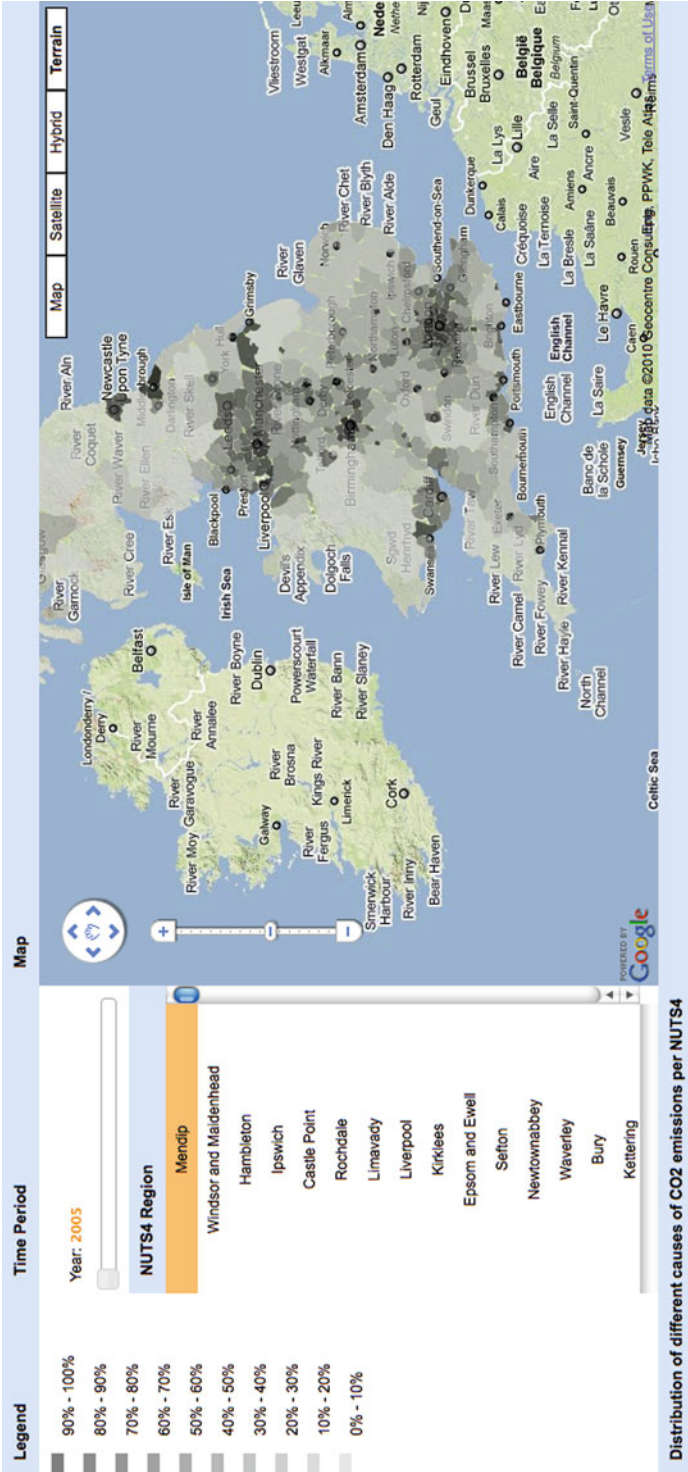


Fig. 20.13

EnAKTiNg UK CO₂ emissions linked data app., eGovernment

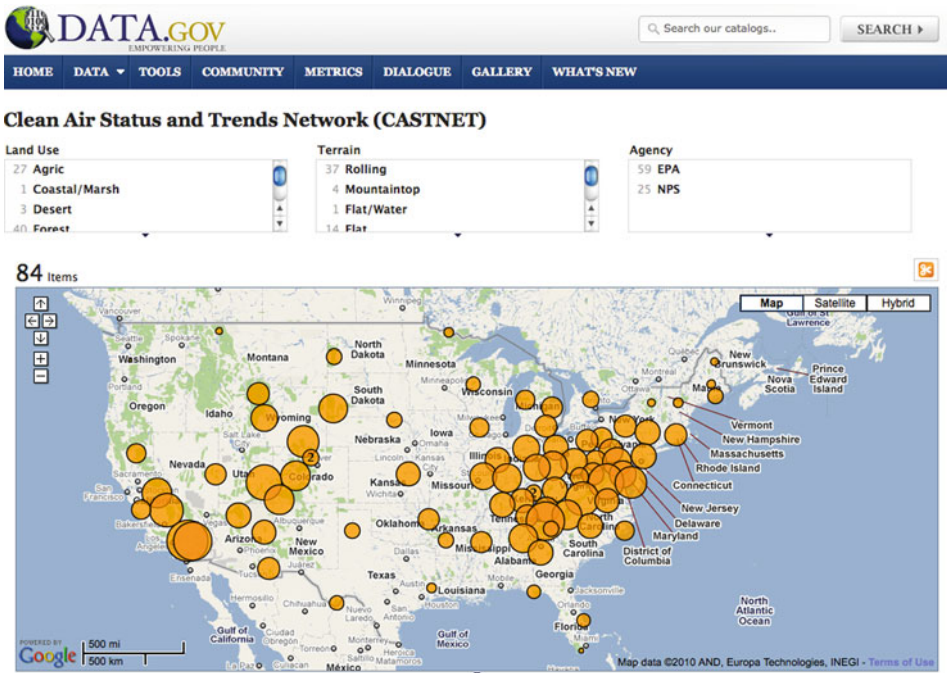


Fig. 20.14
Clean Air Status and Trends Network (CASTNET) application from data.gov, eGovernment

- The EnAKTinG project can be found at <http://www.enaktin.org/>
- SEMIC.EU is at <http://www.semic.eu/semic/view/index.xhtml>
- For OntoGov, see <http://www.hsw.fhso.ch/ontogov/>
- For DIP, see <http://dip.semanticweb.org/>
- The WSMO working group is at <http://www.wsmo.org/>
- Access-eGov is at <http://www.accessegov.org/acegov/web/uk/index.jsp>
- The Tetherless World Constellation is at <http://rpi.edu/research/constellations/tetherlessworld.html>

A very useful volume containing a series of essays describing EU projects, and the general European approach of reengineering is Tomas Vitvar, Vassilios Peristeras, and Konstantinos Tarabanis (eds.), *Semantic Technologies for E-Government*, Berlin: Springer, 2010. It contains papers by most leading Continental European researchers from a range of projects, and broadly covers architectures and process integration, ontologies and interoperability, and portals and user interaction.

20.4 Future Issues

Infrastructure development is an important issue. For the eGovernment domain, the trust layer is all-important, and at the current rate of the SW's progress that is a matter for

research. The description of privacy policies, and discovery of trustworthy services and resources is essential for the future take-up of SW technologies in this space. Nevertheless, it is a common assumption that the SW's reasoning capabilities should allow assessment of indicators of trustworthiness, and of the requirements (e.g., for privacy) of clients.

Other aspects of infrastructure are also lacking. Some of the most promising approaches for supplying services are based around WSMO, but that is still something of a work in progress. However, it has been argued (e.g., [12]) that WSMO has a great deal of potential for supporting eGovernment services.

Full integration of SW technologies and services with the complex environment of eGovernment remains a difficult issue and Gugliotta et al. [12] argue that a complex semantic layer providing a framework explicitly for eGovernment is required. On the other hand, such a complex layer risks adding to the complexity which eGovernment practitioners already perceive. Standards for semantic technologies are important, in order to remove risk from the development process, as well as improving trust in the services provided. The perennial issue of the markup of legacy data, of which of course there is an enormous amount in most government data stores, is also extremely important in this area.

Trust and privacy have been neglected in this field, at least partly because of the requirement to create the functionality able to deal with a complex space. Research in the Policy Aware Web is ongoing, transferring the imperative from curtailing data access to the more tractable one of ensuring fair use of data, via transparency of use and accountability for action (enforcement of privacy policies). Privacy sits alongside, and often in tension with, freedom of information and the need for data protection – for data protection to be properly implemented, a strong recommendation is that users are able to interact with data, by being able to check it for accuracy, amend where necessary, and apply their own privacy policies where this is admissible. Developing systems for presenting data to users, and allowing flexible semantic search, is very important. Once data are out there, other social processes can be brought in to help semanticization – for example, Web 2.0 style tagging, which would allow the harnessing of emergent semantics [9].

Structuring and release of information is a vital early step for increasing data access. AKTive PSI has shown how this does not need heavy-duty ontologies or major and immediate buy-in from a large number of units. Ambition should be reserved for the long run, not the short. Quick wins will be gained from providing the means to link data, rather than applying the full panoply of SW technologies.

But culture change will also be required. Data are often unavailable or hard to process for a number of reasons. First, they can be technically difficult or impossible to access, represented as proprietary databases, inaccessible spreadsheets, or embedded in semi- and unstructured Web pages. Indexes and directories of content are often poor. There is little decentralization and single database models predominate that are not always maintained. Second, organizationally and socially there is little incentive for departments or individuals in them to publish data. Fear of inappropriate release, the lack of standards, the invisibility of the benefits, and fear of disruption to existing legacy systems dominate thinking. Privacy is a particular concern of the cautious here (and sometimes is an excuse

for inertia). Third, licensing and regulatory regimes can add complexity – for instance, the terms of a license might be irrelevant to many aspects of potential reuse.

The lines of a general SW approach to the representation of government data, and making them available either across government departments, or to nongovernmental users (especially citizens) are becoming clear. In more detail, the following are useful and practical steps to ensure that technical standards are met and that institutional culture does not impede progress.

1. Access to data should be supplied using an HTTP browser and SPARQL end points to a single point of online access for public datasets.
2. There should be a standard publication format using agreed W3C linked data standards – ideally using standards adopted by other countries.
3. Lightweight integrative ontologies should be used, and some effort put into selecting and developing common terms where necessary.
4. Copyright and commons standards should be developed that are easy to understand and implement.
5. There should be support for community-based indexing, categorization, and annotation of datasets.
6. There should be support for the exploitation and publication of distributed and decentralized information assets by a wide variety of users.
7. There should be attention to make the above part of departments' routine operations.
8. The information regulatory regime should be adjusted to support the proactive publication of government information.
9. Governments should work to promote international liaison and global standards setting, investing in future international data sharing.
10. Governments should adopt an assumption of total publication for anonymous data using open standards.

These principles currently remain a wish list, with variable application across the range of countries that promote eGovernment. What is encouraging is the increasing signs that countries are looking to follow the examples of the USA and UK.

Much of the success of these endeavors will ultimately depend on political will and leadership. The UK has been fortunate to have successive governments promote and support Government Open Data. Following the formation of the UK Coalition Government in May 2010 Berners-Lee and Shadbolt were asked to join the Public Sector Transparency Board (<http://writetoreply.org/publicsectortransparencyboard/>) whose terms of reference seek to extend and consolidate public rights to data, transparency, and accountability through data release, setting open data standards across the whole public sector and the ongoing development of data.gov.uk. An early output is the set of Public Data Principles (<http://data.gov.uk/blog/new-public-sector-transparency-board-and-public-data-transparency-principles>), one of which states:

Release data quickly, and then re-publish it in linked data form – Linked data standards allow the most powerful and easiest reuse of data. However, most existing internal public

sector data are not in linked data form. Rather than delay any release of the data, our recommendation is to release them “as is” as soon as possible, and then work to convert them to a better format.

These principles are intended to change attitudes and behavior – in this way one can hope to drive culture change in the UK government administration toward an assumption of total publication for anonymous data using open standards. Government information out in the open will drive innovation, as more people are able to interrogate the data for their own purposes. Semantic technologies provide great promise, but pragmatic considerations loom large. It may be that open standards, freely available data, small ontologies, quick wins, and modest models are the way to drive the use of SW in eGovernment, rather than implementing large IT systems entirely from the top down. As has been argued, a judicious mix of top-down and bottom-up strategies is required.

It has been observed that SW technologies are playing a key role in the evolution of eGovernment. In particular, a world of linked open government data offers not just a win for the technology. As the UK’s *Guardian* newspaper wrote in a leading editorial:

- ▶ It is ... hazardous trying to envision how freer data will redraw the boundaries between different communities or recast their relationship with power. But it is reasonable to speculate that the uncovering and unlocking of so much information will drive improvements in public policy. It will level the territory on which voters meet politicians, and could prove a powerful brake on campaigning hyperbole in the coming election. Without the printed word there would have been no informed electorate, no demand for accountability from our leaders – and indeed no democracy at all. Open data will surely revive it, and in time could transform it too [95].

Acknowledgments

This work was supported in part by the EPSRC project EnAKTinG (<http://www.enaktin.org/about.html>), grant no. EP/G008493/1, LiveMemories – Active Digital Memories of Collective Life, Bando Grandi Progetti 2006, Provincia Autonoma di Trento, the EU FET project Living Knowledge (<http://livingknowledge-project.eu/>), contract no. 231126, and the EU project WeGov (<http://wegov-project.eu/>), contract no. 248512. Our thanks to an anonymous reviewer for extremely helpful comments.

References

1. Accenture: eGovernment leadership: High performance, maximum value. http://www.accenture.com/NR/rdonlyres/D7206199-C3D4-4CB4-A7D8-846C94287890/0/gove_egov_value.pdf (2004). Accessed 8 Oct 2010
2. U.K. Cabinet Office: Transformational government: enabled by technology. <http://www.cabinetoffice.gov.uk/media/141734/transgov-strategy.pdf> (2005). Accessed 8 Oct 2010
3. U.S. Office of Management and Budget: Expanding e-government: improved service delivery for the American people using information technology. http://www.whitehouse.gov/omb/budintegration/expanding_egov_2005.pdf (2005)

4. Peristeras, V., Mentzas, G., Tarabanis, K.A., Abecker, A.: Transforming e-government and e-participation through IT. *IEEE Intell. Syst.* **24**(5), 14–19 (2009)
5. Layne, K., Lee, J.: Developing fully functional e-government: a four stage model. *Gov. Inform. Q.* **18**, 122–136 (2001)
6. Berners-Lee, T., Shadbolt, N.: Our manifesto for government data. *Guardian DataBlog*. <http://www.guardian.co.uk/news/datablog/2010/jan/21/timbernslee-government-data> (2010). Accessed 8 Oct 2010
7. Shadbolt, N.: Letter from Professor Nigel Shadbolt to Rt Hon Francis Maude MP, Minister for the Cabinet Office. http://data.gov.uk/sites/default/files/Transparency%20Board%20-%20letter%20from%20Nigel%20Shadbolt%20to%20MCO%2014.06.10.pdp__0.pdf (2010). Accessed 14 June 2010
8. Orthofer, G., Wimmer, M.A.: An ontology for eGovernment: linking the scientific model with concrete projects. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) *Semantic Web Meets e-Government: AAAI Spring Symposium 2006*, Stanford. Technical report SS-06-06, pp. 96–98 (2006)
9. Ilgar, E., Oldenhuizing, J., Mika, P.: Let the citizen speak: a demand-driven e-government portal using semantic web technology. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) *Semantic Web Meets e-Government: AAAI Spring Symposium 2006*, Stanford. Technical report SS-06-06, pp. 54–55 (2006)
10. Klischewski, R.: Semantic web for e-government. In: Traunmüller, R. (ed.) *Electronic Government: Second International Conference (EGOV 2003)*, Prague. *Lecture Notes in Computer Science*, vol. 2739, pp. 288–295. Springer, Berlin (2003)
11. Wagner, C., Cheung, K.S.K., Ip, R.K.F., Böttcher, S.: Building semantic webs for e-government with wiki technology. *Electron. Gov.* **3**, 36–55 (2006)
12. Gugliotta, A., Tanasescu, V., Domingue, J., Davies, R., Gutiérrez-Villarías, L., Rowlatt, M., Richardson, M., Stinčić, S.: Benefits and challenges of applying semantic web services in the e-government domain. In: *Semantics 2006*, Vienna, http://projects.kmi.open.ac.uk/dip/resources/Semantics2006/Semantics2006_DIP_Camera_Ready.pdf (2006)
13. Walden, I.: Privacy and data protection. In: Reed, C., Angel, J. (eds.) *Computer Law: The Law and Regulation of Information Technology*, pp. 459–504. Oxford University Press, Oxford (2007)
14. Alani, H., Kalfoglou, Y., O'Hara, K., Shadbolt, N.: Towards a killer app for the semantic web. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) *Proceedings of the Fourth International Semantic Web Conference (ISWC 2005)*, Galway. *Lecture Notes in Computer Science*, vol. 3729, pp. 829–843. Springer, Berlin (2005)
15. Berners-Lee, T., Hendler, J.A., Lassila, O.: The semantic web. *Sci. Am.* **284**(5), 34–43 (2001)
16. Shadbolt, N., Berners-Lee, T., Hall, W.: The semantic web revisited. *IEEE Intell. Syst.* **21**(3), 96–101 (May–Jun 2006)
17. Berčić, B., Vintar, M.: Ontologies, web services and intelligent agents: ideas for further development of life-event portals. In: Traunmüller, R. (ed.) *Electronic Government: Second International Conference (EGOV 2003)*, Prague. *Lecture Notes in Computer Science*, vol. 2739, pp. 329–334. Springer, Berlin (2003)
18. Corradini, F., Sabucedo, L.Á., Polzonetti, A., Rifón, L.A., Re, B.: A case study of semantic solutions for citizen-centered web portals in eGovernment: the Tecut portal. In: Wimmer, M.A., Scholl, H.J., Grönlund, A. (eds.) *Electronic Government: Sixth International Conference (EGOV 2007)*, Regensburg. *Lecture Notes in Computer Science*, vol. 4656, pp. 204–215. Springer, Berlin (2007)
19. Furdik, K., Klischewski, R., Paralič, M., Sabol, T., Skokan, M.: e-Government service integration and provision using semantic technologies. In: Scholl, H. J., Janssen, M., Traunmüller, R., Wimmer, M. A. (eds.) *Electronic Government. Proceedings of Ongoing Research, General Development Issues and Projects of Electronic Government: Eighth International Conference (EGOV 2009)*, Linz, pp. 273–280. http://web.tuke.sk/fei-cit/furdik/publik/egov09_aeg.pdf (2009)
20. Furdik, K., Sabol, T., Bednar, P.: Framework for integration of eGovernment services on a semantic basis. Presented at *Electronic Government: Sixth International Conference (EGOV 2007)*, Regensburg. *Lecture Notes in Computer Science*, vol. 4656. Springer, Berlin. <http://www.accessegov>.

org/acegov/uploadedFiles/webfiles/cfile_10_29_07_9_50_10_AM.pdf (2007)

21. Skokan, M., Bednár, P.: Semantic orchestration of services in e-government. In: Znalosti 2008, Bratislava, pp. 215–223. <http://znalosti2008.fit.stuba.sk/download/articles/znalosti2008-Bednar.pdf> (2008)
22. Maltese, V., Giunchiglia, F., Denecke, K., Lewis, P., Wallner, C., Baldry, A., Madalli, D.: On the interdisciplinary foundations of diversity. Dipartimento di Ingegneria e Scienza dell'Informazione, University of Trento. Technical report DISI-09-040. <http://eprints.biblio.unitn.it/archive/00001651/01/040.pdf> (2009)
23. Klischewski, R.: Top down or bottom up? How to establish a common ground for semantic interoperability within e-government communities. In: Traunmüller, R., Palmirani, M. (eds.) e-Government: Modelling Norms and Concepts as Key Issues. Proceedings of the First International Workshop on e-Government at ICAIL 2003, Edinburgh (2003)
24. Klischewski, R.: Information integration or process integration? How to achieve interoperability in administration. In: Traunmüller, R. (ed.) Electronic Government: Third International Conference (EGOV 2004), Zaragoza. Lecture Notes in Computer Science, vol. 3183, pp. 57–65. Springer, Berlin (2004)
25. Guijarro, L.: Semantic interoperability in eGovernment initiatives. *Comput. Stand. Inter.* **31**, 174–180 (2009)
26. Mach, M., Sabol, T., Paralic, J.: Integration of eGov services: back-office versus front-office integration. In: Abecker, A., Mentzas, G., Stojanovic, L. (eds.) Proceedings of the Workshop on Semantic Web for eGovernment 2006, Workshop at the Third European Semantic Web Conference (ESWC 2006), Budva. Lecture Notes in Computer Science, vol. 4011, pp. 48–53. Springer, Berlin. <http://www.imu.iccs.gr/semgov/final/SemanticWebForEGovernement-Proceeding.pdf> (2006)
27. Weinstein, P.: Model unification in support of political process. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) Semantic Web Meets e-Government: AAAI Spring Symposium 2006, Stanford. Technical report SS-06-06, pp. 138–140 (2006)
28. Della Valle, E., Cerizza, D., Celino, I., Estublier, J., Vega, G., Kerrigan, M., Ramírez, J., Villazon, B., Guarrera, P., Zhao, G., Monteleone, G.: SEEMP: a semantic interoperability infrastructure for e-government services in the employment sector. In: Franconi, E., Kifer, M., May, W. (eds.) The Semantic Web: Research and Applications: Proceedings of the Fourth European Semantic Web Conference (ESWC 2007), Innsbruck. Lecture Notes in Computer Science, vol. 4519, pp. 220–234. Springer, Berlin (2007)
29. Herborn, T., Wimmer, M.A.: Process ontologies facilitating interoperability in e-government: a methodological framework. In: Hinkelmann, K., Karagiannis, D., Stojanovic, N., Wagner, G. (eds.) Proceedings of the Workshop on Semantics for Business Process Management: Third European Semantic Web Conference (ESWC 2006), Budva. Lecture Notes in Computer Science, vol. 4011, pp. 76–88. Springer, Berlin (2006)
30. Van Elst, L., Klein, B., Maus, H., Schöning, H., Tommasi, A., Zavattari, C., Favaro, J., Giannella, V.: Business register interoperability throughout Europe: BRITE. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) Semantic Web Meets e-Government: AAAI Spring Symposium 2006, Stanford. Technical report SS-06-06, pp. 70–79 (2006)
31. Fraser, J., Adams, N., Macintosh, A., McKay-Hubbard, A., Lobo, T.P., Pardo, P.F., Martínez, R.C., Vallecillo, J.S.: Knowledge management applied to e-government services: the use of an ontology. In: Wimmer, M.A. (ed.) Knowledge Management in Electronic Government: Proceedings of Fourth IFIP International Working Conference (KMGov 2003), Rhodes, pp. 116–126. Springer, Berlin (2003)
32. Barnickel, N., Fluegge, M., Schmidt, K.-U.: Interoperability in eGovernment through cross-ontology Semantic Web service composition. In: Abecker, A., Mentzas, G., Stojanovic, L. (eds.) Proceedings of the Workshop on Semantic Web for eGovernment: Third European Semantic Web Conference (ESWC 2006), Budva. Lecture Notes in Computer Science, vol. 4011, pp. 37–47. Springer, Berlin (2006)
33. Maedche, A., Motik, B., Silva, N., Volz, R.: MAFRA – A Mapping FRamework for distributed ontologies. In: Gómez-Pérez, A., Benjamins, V.R. (eds.) Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web:

- 13th International Conference (EKAW 2002), Sigvenza, pp. 235–250. Springer, Berlin (2002)
34. Czajkowski, M., Ashpole, B., Hughes, T., Le, T.: Bridging semantic eGovernment applications using ontology-to-ontology message translation. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) *Semantic Web Meets e-Government: AAAI Spring Symposium 2006*, Stanford. Technical report SS-06-06, pp. 4–6 (2006)
 35. Salvadores, M., Correndo, G., Szomszor, M., Yang, Y., Gibbins, N., Millard, I., Glaser, H., Shadbolt, N.: Domain-specific backlinking services in the web of data. In: *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI 2010)*, Toronto. <http://eprints.ecs.soton.ac.uk/21258/> (2010)
 36. Breslin, J.G., Harth, A., Bojars, U., Decker, S.: Towards semantically-interlinked online communities. In: Gómez-Pérez, A., Euzenat, J. (eds.) *The Semantic Web: Research and Applications: Second European Semantic Web Conference (ESWC 2005)*, Heraklion. *Lecture Notes in Computer Science*, vol. 3532, pp. 500–514. Springer, Berlin (2005)
 37. Tullo, C.: Information age. Civil service network. <http://www.civilservicenetwork.com/features/features-article/newarticle/information-age/> (2008). Accessed 24 Oct 2008
 38. Pullinger, D.: Building the public sector information infrastructure. Talk presented at Conference on Unlocking the Potential of Public Sector Information, London. http://www.epsiplus.net/media/files/3_14october2008_david_pullinger (2008)
 39. Alani, H., Dupplaw, D., Sheridan, J., O’Hara, K., Darlington, J., Shadbolt, N., Tullo, C.: Unlocking the potential of public sector information with semantic web technology. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) *The Semantic Web: Sixth International Semantic Web Conference, Second Asian Semantic Web Conference, ISWC 2007 + ASWC 2007*, Busan. *Lecture Notes in Computer Science*, vol. 4825, pp. 708–721. Springer, Berlin (2007)
 40. U.K. Office of Public Sector Information: The United Kingdom implementation of the European directive on the re-use of public sector information – the first two years. <http://www.opsi.gov.uk/advice/psi-regulations/uk-implementation-first-years.pdf> (2007)
 41. Comte, F., Leclère, M.: A semantical reasoning framework for eGovernment of the French social welfare system. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) *Semantic Web Meets e-Government: AAAI Spring Symposium 2006*, Stanford. Technical report SS-06-06, pp. 7–9 (2006)
 42. Sidoroff, T., Hyvönen, E.: Semantic e-government portals – a case study. In: *Proceedings of Semantic Web Case Studies and Best Practice for eBusiness (SWCASE 2005)*, International Semantic Web Conference (ISWC 2005), Galway. <http://www.seco.tkk.fi/publications/2005/sidoroff-hyvonen-semantic-e-government-2005.pdf> (2005)
 43. Peristeras, V., Goudos, S.K., Loutas, N., Tarabanis, K.: Ontology-based search for eGovernment services using citizen profile information. *J. Web Eng.* **8**(3), 245–267 (2009)
 44. Markellos, K., Sakkopoulos, E., Sirmakessis, S., Tsakalidis, A.: Semantic web search for e-government: the case study of Intrastat. *J. Internet Technol.* **8**, 1–12 (2007)
 45. Apostolou, D., Stojanovic, N., Anicic, D.: Responsive knowledge management for public administration: an event-driven approach. *IEEE Intell. Syst.* **24**(5), 20–30 (2009)
 46. Stojanovic, N., Mentzas, G., Apostolou, D.: Semantic-enabled agile knowledge-based e-government. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) *Semantic Web Meets e-Government: AAAI Spring Symposium 2006*, Stanford. Technical report SS-06-06, pp. 132–134 (2006)
 47. Sacco, G.M.: User-centric access to e-government information: e-citizen discovery of e-services. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) *Semantic Web Meets e-Government: AAAI Spring Symposium 2006*, Stanford. Technical report SS-06-06, pp. 114–116 (2006)
 48. Medjahed, B., Bouguettaya, A.: Customized delivery of e-government web services. *IEEE Intell. Syst.* **20**(6), 77–84 (2005)
 49. Bouguettaya, A., Medjahed, B., Rezgui, A., Ouzzani, M., Liu, X., Yu, Q.: WebDG – a platform for e-government web services. In: Wang, S., Tanaka, K., Zhou, S., Ling, T.W., Guan, J., Yang, D., Grandi, F., Mangina, E., Song, I.-Y., Mayr, H.C. (eds.) *Conceptual Modeling for Advanced Application Domains: ER2004*

- Workshops CoMoGIS, CoMWIM, EDCM, CoMoA, DGOV and eCOMO, pp. 553–565. Springer, Berlin (2004)
50. Chun, S.A., Atluri, V., Adam, N.R.: Policy-based Web service composition. In: Proceedings of the 14th International Workshop on Research Issues in Data Engineering: Web Services for e-Commerce and e-Government Applications (RIDE WS-ECEG 2004), Boston, pp. 85–92 (2004)
 51. Medjahed, B., Bouguettaya, A., Elmagarmid, A.K.: Composing web services on the semantic web. *Vldb J.* **12**, 333–351 (2003)
 52. Medjahed, B., Rezgui, A., Bouguettaya, A., Ouzzani, M.: Infrastructure for e-government web services. *IEEE Internet Comput.* **7**(1), 58–65 (2003)
 53. Cabral, L., Domingue, J., Galizia, S., Gugliotta, A., Tanasescu, V., Pedrinaci, C., Norton, B.: IRS-III: A broker for Semantic Web services based applications. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, M. (eds.) Proceedings of the Fifth International Semantic Web Conference (ISWC 2006), Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 201–214. Springer, Berlin (2006)
 54. Gugliotta, A., Domingue, J., Cabral, L., Tanasescu, V., Galizia, S., Davies, R., Gutierrez Villarias, L., Rowlatt, M., Richardson, M., Stincic, S.: Deploying semantic web services-based applications in the e-government domain. *J. Data Semant.* **10**, 96–132 (2008)
 55. Gugliotta, A., Cabral, L., Domingue, J., Roberto, V., Rowlatt, M., Davies, R.: A semantic web service-based architecture for the interoperability of e-government services. In: Proceedings of the Web Information Systems Modeling Workshop (WISM 2005), Fifth International Conference on Web Engineering (ICWE 2005), Sydney. <http://oro.open.ac.uk/3005/1/SemanticWeb.pdf> (2005)
 56. Wang, X., Vitvar, T., Peristeras, V., Mocan, A., Goudos, S.K., Tarabanis, K.: WSMO-PA: Formal specification of public administration service model on semantic web service ontology. In: Proceedings of the 40th Annual Hawaii International Conference on System Science (HICSS 2007), Waikoloa. <http://portal.acm.org/citation.cfm?id=1255783> (2007)
 57. Peristeras, V., Tarabanis, K.: Providing pan-European e-government services with the use of semantic web services technologies: a generic process model. In: Wimmer, M.A., Traummüller, R., Grönlund, Å., Andersen, K.V. (eds.) Electronic Government: Fourth International Conference (EGOV 2005), Copenhagen. Lecture Notes in Computer Science, vol. 3591, pp. 226–236. Springer, Berlin (2005)
 58. Peristeras, V., Tarabanis, K.: Reengineering public administration through semantic technologies and a reference domain ontology. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) Semantic Web Meets e-Government: AAAI Spring Symposium 2006, Stanford. Technical report SS-06-06, pp. 99–110 (2006)
 59. O'Hara, K., Shadbolt, N.: *The Spy in the Coffee Machine: The End of Privacy As We Know It*. Oneworld, Oxford (2008)
 60. Klischewski, R., Jeenicke, M.: Semantic Web technologies for information management within e-government services. In: Proceedings of the 37th Annual Hawaii International Conference on System Science (HICSS 2004), Waikoloa. <http://www2.computer.org/portal/web/csdl/doi/10.1109/HICSS.2004.1265305> (2004)
 61. Weitzner, D.J., Abelson, H., Berners-Lee, T., Hanson, C., Hendler, J., Kagal, L., McGuinness, D.L., Sussman, G.J., Waterman, K.K.: Transparent accountable data mining: new strategies for privacy protection. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) Semantic Web Meets e-Government: AAAI Spring Symposium 2006, Stanford. Technical report SS-06-06, pp. 141–150 (2006)
 62. Stojanovic, L., Abecker, A., Stojanovic, N., Studer, R.: On managing changes in the ontology-based e-government. In: Meersman, R., Tari, Z., van der Aalst, W., Bussler, C., Gal, A., Cahill, V., Vinoski, S., Vogels, W., Catarci, C., Sycara, K. (eds.) On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, ODBASE, vol. 2, pp. 1080–1097. Springer, Berlin (2004)
 63. Klischewski, R., Ukena, S.: Designing semantic e-government services driven by user requirements. Presented at Electronic Government: Sixth International Conference (EGOV 2007), Regensburg. Lecture Notes in Computer Science, vol. 4656. Springer, Berlin. http://www.accessegov.org/acegov/uploadedFiles/webfiles/cffile_10_29_07_9_46_36_AM.pdf (2007)
 64. Stojanovic, L., Abecker, A., Apostolou, D., Mentzas, G., Studer, R.: The role of semantics

- in e-government service model verification and evolution. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) *Semantic Web Meets e-Government: AAAI Spring Symposium 2006*, Stanford. Technical report SS-06-06, 117–128 (2006)
65. Stojanovic, L., Stojanovic, N., Apostolou, D.: Change management in e-government: ontoGov case study. *Electron. Gov.* **3**, 74–92 (2006)
 66. Hinkelmann, K., Probst, F., Thönssen, B.: Agile process management: framework and methodology. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) *Semantic Web Meets e-Government: AAAI Spring Symposium 2006*, Stanford. Technical report SS-06-06, pp. 33–41 (2006)
 67. Stojanovic, N., Stojanovic, L., Hinkelmann, K., Mentzas, G., Abecker, A.: Fostering self-adaptive e-government service improvement using semantic technologies. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) *Semantic Web Meets e-Government: AAAI Spring Symposium 2006*, Stanford. Technical report SS-06-06, pp. 129–131 (2006)
 68. Apostolou, D., Stojanovic, L., Lobo, T.P., Thoenssen, B.: Towards a semantically-driven software engineering environment for eGovernment. In: Böhlen, M., Gamper, J., Polasek, W., Wimmer, M.A. (eds.) *e-Government: Towards Electronic Democracy: Proceedings of the International Conference (TCGOV 2005)*, Bolzano. Lecture Notes in Computer Science, vol. 3416, pp. 157–168. Springer, Berlin (2005)
 69. Tambouris, E., Gorilas, S., Kavadias, G., Apostolou, D., Abecker, A., Stojanovic, L., Mentzas, G.: Ontology-enabled e-gov service configuration: an overview of the OntoGov project. In: Wimmer, M.A. (ed.) *Knowledge Management in Electronic Government: Proceedings of the Fifth IFIP International Working Conference (KMGov 2004)*, Krams, pp. 122–127. Springer, Berlin (2004)
 70. Fishenden, J., Bell, O., Grose, A.: Government interoperability: enabling the delivery of e-services. Microsoft white paper (2005)
 71. Koné, M.T., Jaafar, F.B., Saïd, A.M.: A critical step in eGovernment evolution. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) *Semantic Web Meets e-Government: AAAI Spring Symposium 2006*, Stanford. Technical report SS-06-06, pp. 64–69 (2006)
 72. Bennett, D., Harvey, A.: Publishing open government data, W3C Working Draft. <http://www.w3.org/TR/gov-data/> (Sept 2009). Accessed 8 Oct 2010
 73. Klischewski, R.: Migrating small governments' websites to the semantic web. In: Abecker, A., Sheth, A., Mentzas, G., Stojanovic, L. (eds.) *Semantic Web Meets e-Government: AAAI Spring Symposium 2006*. Technical report SS-06-06, pp. 56–63, Stanford (2006)
 74. Alani, H., Hall, W., O'Hara, K., Shadbolt, N., Chandler, P., Szomszor, M.: Building a pragmatic semantic web. *IEEE Intell. Syst.* **23**(3), 61–68 (2008)
 75. Berners-Lee, T.: The fractal nature of the web. <http://www.w3.org/DesignIssues/Fractal.html> (1998/2006). Accessed 8 Oct 2010
 76. U.K. Office of Public Sector Information: Unlocking PSI potential: the United Kingdom report on the re-use of public sector information. <http://www.opsi.gov.uk/advice/psi-regulations/uk-report-reuse-psi-2008.pdf> (2008)
 77. Bizer, C., Cyganiak, R., Heath, T.: How to publish linked data on the web. <http://sites.wiwiw.fu-berlin.de/suhl/bizer/pub/LinkedDataTutorial/> (2007)
 78. Berners-Lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J., Lerer, A., Sheets, D.: Tabulator: exploring and analyzing linked data on the semantic web. In: *Third International Semantic Web User Interaction Workshop (SWUI 2006)*, Athens. <http://swui.semanticweb.org/swui06/papers/Berners-Lee/Berners-Lee.pdf> (2006)
 79. Berners-Lee, T.: Linked data. <http://www.w3.org/DesignIssues/LinkedData.html> (2007/2009). Accessed 8 Oct 2010
 80. Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. *Knowl. Eng. Rev.* **18**(1), 1–31 (2003)
 81. Bizer, C., Heath, T., Berners-Lee, T.: Linked data – The story so far. In: Heath, T., Hepp, M., Bizer, C. (eds.) *Int. J. Semant. Web Inf. Syst.* **5**(3), 1–22, (2009). Special issue on linked data. <http://tomheath.com/papers/bizer-heath-berners-lee-ijswis-linked-data.pdf>
 82. Acar, S., Alonso, J.M., Novak, K.: Improving access to government through better use of the web. <http://www.w3.org/TR/egov-improving/> (2009). Accessed 8 Oct 2010
 83. Crabtree, J.: Whitehall's web revolution: the inside story. Prospect, 166. <http://www>

prospectmagazine.co.uk/2010/01/whitehalls-web-revolution-the-inside-story/ (2010)

84. T, J.: Calling open data developers – we need your help. Cabinet Office Digital Engagement Blog. <http://blogs.cabinetoffice.gov.uk/digitalengagement/post/2009/09/30/Calling-Open-Data-Developers-We-need-your-help.aspx> (2009)
85. Arthur, C.: UK's free data website 'is a world showcase'. Guardian DataBlog. <http://www.guardian.co.uk/technology/datablog/2010/jan/21/government-free-data-website-launch> (2010). Accessed 8 Oct 2010
86. Berners-Lee, T.: Tim Berners-Lee on the next web, TED talk. http://www.ted.com/talks/tim_berners_lee_on_the_next_web.html (2009)
87. Sheridan J, Tennison J.: Linking UK government data. In: Proceedings of the Linked Data on the Web (LDOW 2010) Workshop at WWW 2010, Raleigh. <http://events.linkeddata.org/ldow2010/slides/ldow2010-slides-tennison.pdf> (2010)
88. Chief Technology Officer Council: Designing URI sets for the UK public sector. http://www.cabinetoffice.gov.uk/media/301253/puublic_sector_uri.pdf (2009)
89. Sauer mann L., Cyganiak R.: Cool URIs for the semantic web, W3C Interest Group Note 03. <http://www.w3.org/TR/cooluris/> (Dec 2008)
90. Carroll, J.J., Bizer, C., Hayes, P.J., Stickler, P.: Named graphs, provenance and trust. In: Proceedings of the 14th International Conference on World Wide Web (WWW 2005), Chiba. pp. 613–622 (2005)
91. Baker, T.: Maintaining Dublin core as a semantic web vocabulary. In: Matthias, H., Claudia, N., Thomas, R. (eds.), From Integrated Publication and Information Systems to Virtual Information and Knowledge Environments: Essays Dedicated to Erich J. Neuhold on the Occasion of his 65th Birthday, pp. 61–68. Springer, Berlin (2005)
92. Moreau, L.: The foundations for provenance on the web. *Found. Trend Web Sci.* 2(2) (2010)
93. Cyganiak, R., Field, S., Gregory, A., Halb, W., Tennison, J.: Semantic statistics: bringing together SDMX and SCOVO. In: Proceedings of the Linked Data on the Web (LDOW 2010) Workshop at WWW 2010, Raleigh (2010)
94. Omitola, T., Koumenides, C. L., Popov, I. O., Yang, Y., Salvadores, M., Szomszor, M., Berners-Lee, T., Gibbins, N., Hall, W., Schraefel, M., Shadbolt, N. (2010) Put in your postcode, out comes the data: a case study. In: Proceedings of the Seventh Extended Semantic Web Conference (ESWC 2010), Heraklion. *Lecture Notes in Computer Science*, vol. 6088, pp. 318–332. Springer, Berlin (2010)
95. The Guardian: Government information: creative commons. Guardian newspaper leader column. <http://www.guardian.co.uk/commentisfree/2010/jan/23/government-information-creative-commons-internet> (2010)

21 Multimedia, Broadcasting, and eCulture

Lyndon Nixon¹ · Stamatia Dasiopoulou² · Jean-Pierre Evain³ · Eero Hyvönen⁴ · Ioannis Kompatsiaris² · Raphaël Troncy⁵

¹Semantic Technology Institute (STI) International, Vienna, Austria

²Centre for Research and Technology Hellas (CERTH), Themi-Thessaloniki, Greece

³European Broadcasting Union (EBU), Grand-Saconnex, Switzerland

⁴Aalto University and University of Helsinki, Aalto, Finland

⁵EURECOM, Sophia Antipolis, France

21.1	<i>Introduction</i>	913
21.2	<i>Scientific and Technical Overview</i>	915
21.2.1	Multimedia Semantics: Vocabularies and Tools	915
21.2.1.1	Multimedia Vocabularies on the Semantic Web	915
21.2.2	Semantic Web-Based Multimedia Annotation Tools	923
21.2.3	Semantic Multimedia Analysis	929
21.2.4	Semantics in Broadcasting	935
21.2.4.1	Metadata in Broadcasting from Its Origin	935
21.2.4.2	Metadata Standardization in Broadcasting	936
21.2.4.3	Using Ontologies: Metadata + Semantic	937
21.2.4.4	A Semantic Representation of TV-Anytime in a Nutshell	937
21.2.4.5	A Semantic Representation of Thesauri	941
21.2.4.6	The Holy Grail: Agreeing on a Class Model	942
21.2.4.7	Agreeing on Properties	943
21.3	<i>Example Applications</i>	945
21.3.1	Semantic Television	945
21.3.1.1	User Activity Capture	947
21.3.1.2	Enriched EPG Data	948
21.3.1.3	Alignment Between Vocabularies	948
21.3.1.4	Personalized TV Program Recommendation	949
21.3.2	Semantics in Cultural Heritage	949
21.3.2.1	Ontological Dimensions	951
21.3.2.2	Challenges of Content Creation	953

21.3.2.3	Syntactic and Semantic Interoperability	954
21.3.2.4	Semantic eCulture Systems	955
21.3.2.5	Semantic Search	959
21.3.2.6	Semantic Browsing and Recommending	960
21.3.2.7	Visualization	961
21.3.2.8	Cultural Heritage as Web Services	962
21.4	<i>Related Resources Including Key Papers</i>	962
21.4.1	Multimedia Ontologies, Annotation, and Analysis	962
21.4.2	Broadcaster Artifacts Online	963
21.4.2.1	Vocabularies and Ontologies	963
21.4.2.2	Metadata Schemas	964
21.4.2.3	Semantic Television	964
21.4.3	Cultural Heritage Artifacts Online	964
21.4.3.1	Vocabularies and Ontologies	964
21.4.3.2	Metadata Schemas	965
21.4.3.3	Semantic eCulture Systems Online	965
21.5	<i>Future Issues</i>	965
21.6	<i>Cross-References</i>	969

Abstract: This chapter turns to the application of semantic technologies to areas where text is not dominant, but rather audiovisual content in the form of images, 3D objects, audio, and video/television. Non-textual digital content raises new challenges for semantic technology in terms of capturing the meaning of that content and expressing it in the form of semantic annotation. Where such annotations are available in combination with expressive ontologies describing the target domain, such as television and cultural heritage, new and exciting possibilities arise for multimedia applications.

21.1 Introduction

Ever since computers became capable of processing data other than text, capturing, storing and processing images, 3D modeling, and processing audio and video, the issues of how to describe these data so that they could be found again, or process these data so that they could be reused in new contexts, have been studied in the field of multimedia systems. The subject of this chapter is the work emerging on the intersection of multimedia systems and semantic technology, leading to new insights in multimedia analysis and annotation, and by extension new applications in areas like broadcasting (television) and cultural heritage.

General cross-media queries are textual in nature, as text is considered the easiest media for a computer system to handle. In order that queries are then matched to media, the media objects are manually textually annotated. Then established text matching algorithms are applicable to the multimedia retrieval. This additional annotation of data is often referred to as “metadata,” which means “data about data.” In annotated systems, how the user forms the query can be very significant in determining the success of the retrieval, both in terms of the ambiguity of natural language and that the user may be unaware of how the media has been annotated. Annotated systems are also not aware of the broader meaning of the terms used in their metadata vocabulary, for example, that the keyword “Ford Orion” is a specific instance of a “car,” which is a “vehicle.” Hence, retrieval is rather coarse, for example, only media with the exact annotation searched for is returned, rather than with other, similar, media. To overcome this, text-based approaches such as Latent Semantic Indexing [1] analyze natural language and associate related words. This type of approach is still very dominant on the Web, for example, Google Image Search (possibly the most used image retrieval system on the Web at the time of writing) associates images with the text closest to them on the HTML page. In all these cases, the metadata are determinable only as a result of there already being natural language text associated with the media.

The set of semantic technologies addressed previously offers a new solution to the problems of multimedia retrieval and processing. Multimedia annotations can become richer than just simple metadata with keywords, where the use of ontologies enables the annotator to link annotation values to knowledge about the wider domain, whether that domain is that of the media object’s representation (e.g., a picture of a Ford Orion linked

into an ontology about cars) or of the media object itself (e.g., metadata on an art painting is described in terms of an ontology about art paintings, which can capture domain knowledge about paintings such as the materials used, style employed, etc.). Hence, the choice of an appropriate semantic schema to annotate multimedia content is important with respect to its future (re)use and most multimedia schemas in use today are not immediately usable together with ontologies and reasoners.

➤ [Section 21.2.1](#) begins with an exploration of semantic multimedia with the current status of multimedia ontologies for annotation and the work toward a shared Media Ontology. This is complemented by an overview of current tools for multimedia annotation in ➤ [Sect. 21.2.2](#).

Multimedia content selection is a very different and difficult problem in comparison with textual retrieval where the query is usually also textual and is realized by string matching in the content store, aided by devices such as stemming and synonyms. The key problems in the case of multimedia retrieval are that the form of query does not generally match the form of media being queried and with queries that are of the same form (e.g., user whistling to search an audio database) matching techniques are more complex than with text. However, in media industries, it is more typical to search image data on the basis of an existing image, or audio based on a note or sample. Here, MIR (multimedia information retrieval) research to improve the so-called query by example focuses on low-level feature extraction and developing classifiers that map these low-level features to a high-level concept. However, such low-level matching has the restriction of requiring the query to be in the same form as the stored media, and conversely, that the stored media is all of a single form. Hence, mixed media stores are excluded from this approach, and queries are often not intuitive to the general user (e.g., much depends on the user's skill for drawing or whistling). The use of such classification techniques to support the multimedia annotation not only helps reduce human annotation effort but provides means for cross-media multimedia search, or even mixed-form queries (query by example with identification of the concepts sought, to better rank or filter results). ➤ [Section 21.2.3](#) introduces semantic multimedia analysis techniques to better train the classifiers and extract concepts from low-level features.

The broadcasting industry relies on schemas and standards for its metadata, and a look into developments toward a semantic schema standard for broadcasters in the future is provided in ➤ [Sect. 21.2.4](#).

In turn, semantic data about multimedia objects allow them to be processed and manipulated in similar ways to other instance data, for example, SPARQL-based retrieval of matching objects, data mediation to ensure interoperability of schemas across systems, or transformations to effect adaptation of the described media object to new devices or contexts. In the rest of this chapter, in ➤ [Sect. 21.3](#), applications of semantic technologies applied and adapted to the multimedia domain are presented, with examples from the broadcasting (➤ [Sect. 21.3.1](#)) and cultural heritage (➤ [Sect. 21.3.2](#)) sectors, respectively.

After providing some key papers on the current work in semantic multimedia (➤ [Sect. 21.4](#)), ➤ [Sect. 21.5](#) will turn to the future trends in this area, considering

particularly how future TV viewers and explorers of 3D virtual worlds may benefit from today's research and enjoy the use of semantic technologies without being aware of it.

21.2 Scientific and Technical Overview

Before turning to applications of semantic multimedia and its future in society and industry, it is necessary to introduce the current state of the art of the building blocks of semantic multimedia: firstly, the vocabularies that are formalized as ontologies and used to semantically describe multimedia content. Then the means to creating those annotations, both through manual editing in tools and through automated generation using multimedia analysis techniques. Finally, we look at how the state of the art in the broadcasting industry is moving toward the use of semantic technology.

21.2.1 Multimedia Semantics: Vocabularies and Tools

The availability of interoperable semantic metadata is crucial for handling effectively the growing amount of multimedia assets that are encountered in a plethora of applications addressing both personal and professional multimedia data usage. A growing adoption of Semantic Web technologies by the multimedia community in order to enable large-scale interoperability between media descriptions and to benefit from the advantages brought by explicit semantics in the reuse, sharing, and processing of metadata can be observed.

Multimedia annotations present several challenges. One of them is to enable users to describe the content of some assets with respect to specific domain ontologies, but contrary to the annotation of textual resources, multimedia content does not contain canonical units (similar to words) that would have a predefined meaning. In the case of media annotation, particular requirements apply as a result of the intrinsically multidisciplinary nature of multimedia content. Among the most fundamental of these is the ability to localize and annotate specific subparts within a given media asset, such as regions in a still image or moving objects in video sequences. The modeling of the structural and decomposition knowledge involved in the localization of individual media segments varies across vocabularies and has different levels of support among the existing annotation tools. The supported types of metadata, the granularity and expressivity of the annotation level, the intended context of usage, etc., give rise to further differences, casting a rather obscure setting regarding the sharing and reuse of the generated multimedia annotations.

21.2.1.1 Multimedia Vocabularies on the Semantic Web

There has been a proliferation of metadata formats to express information about media objects. For example, pictures taken by camera come with EXIF metadata related to the

image data structure (height, width, orientation), the capturing information (focal length, exposure time, flash), and the image data characteristics (transfer function, color space transformation). These technical metadata are generally completed with other standards aiming at describing the subject matter. DIG35 is a specification of the International Imaging Association (I3A). It defines, within an XML Schema, metadata related to image parameters, creation information, content description (who, what, when, and where), history, and intellectual property rights. XMP provides a native RDF data model and predefined sets of metadata property definitions such as Dublin Core, basic rights, and media management schemas for describing still images. IPTC has itself integrated XMP in its Image Metadata specifications.

Video can be decomposed and described using MPEG-7, the Multimedia Content Description ISO Standard. This language provides a large and comprehensive set of descriptors including multimedia decomposition descriptors, management metadata properties, audio and visual low-level features, and more abstract semantic concepts. From the broadcast world, the European Broadcaster Union (EBU) has actively contributed to the video extension of the new version of IPTC NewsML-G2 based on IPTC's NAR architecture for describing videos, providing some extensions in order to be able to associate metadata to arbitrary parts of videos and to have a vocabulary for rights management. The EBU has also developed the EBUCore, P-Meta, and TV-Anytime standards for production, archives, and electronic program guides (EPG). Finally, video-sharing platforms provide generally their own lightweight metadata schemas and APIs such as Yahoo!, Media RSS, or Google Video sitemaps.

Many of these formats are further described and discussed in [2]. On the one hand, an environment is observed that uses numerous languages and formats, often XML-based, that leads to interoperability problems and that excludes linking to other vocabularies and existing Web knowledge resources. On the other hand, there is a need for using and combining some of these metadata formats on the Web and there has been research work for enabling interoperability using Semantic Web technologies. The following first describes the various attempts to bring the most famous standard, MPEG-7, into the Semantic Web. Then an ontology is presented for media resources that aims to be a future W3C recommendation.

Comparing Four Different MPEG-7 Ontologies

MPEG-7, formally named *Multimedia Content Description Interface* [3], is an ISO/IEC standard developed by the Moving Picture Experts Group (MPEG) for the structural and semantic description of multimedia content. MPEG-7 standardizes *tools* or ways to define multimedia *Descriptors* (Ds), *Description Schemes* (DSs), and the relationships between them. The descriptors correspond either to the data features themselves, generally low-level features such as visual (e.g., texture, camera motion) and audio (e.g., spectrum, harmony), or semantic objects (e.g., places, actors, events, objects). Ideally, most low-level descriptors would be extracted automatically, whereas human annotation would be required for producing high-level descriptors. The description schemes are used for grouping the descriptors into more abstract description entities. These tools as well as

their relationships are represented using the *Description Definition Language* (DDL). After a requirement specification phase, the W3C XML Schema recommendation has been adopted as the most appropriate syntax for the MPEG-7 DDL.

The flexibility of MPEG-7 is therefore based on allowing descriptions to be associated with arbitrary multimedia segments, at any level of granularity, using different levels of abstraction. The downside of the breadth targeted by MPEG-7 is its complexity and its ambiguity. Hence, MPEG-7 XML Schemas define 1,182 elements, 417 attributes, and 377 complex types, which make the standard difficult to manage. Moreover, the use of XML Schema implies that a great part of the semantics remains implicit. For example, very different syntactic variations may be used in multimedia descriptions with the same intended semantics, while remaining valid MPEG-7 descriptions. Given that the standard does not provide a formal semantics for these descriptions, this syntax variability causes serious interoperability issues for multimedia processing and exchange [4–6]. The profiles introduced by MPEG-7 and their possible formalization [7] concern, by definition, only a subset of the whole standard. For alleviating the lack of formal semantics in MPEG-7, four multimedia ontologies represented in OWL and covering the whole standard have been proposed (see Table 21.1) [8–10]. The proposers of these four ontologies have compared and discussed these four modeling approaches [11]. First, these four ontologies are briefly described and then their commonalities and differences are outlined using three criteria: (1) the way the multimedia ontology is linked with domain semantics; (2) the MPEG-7 coverage of the multimedia ontology; and (3) the scalability and modeling rationale of the conceptualization.

In 2001, Hunter proposed an initial manual translation of MPEG-7 into RDFS (and then into DAML+OIL) and provided a rationale for its use within the Semantic Web [9]. This multimedia ontology was translated into OWL, and extended and harmonized using the ABC upper ontology [12] for applications in the digital libraries [13] and eResearch fields [14]. The current version is an OWL Full ontology containing classes defining the media types (Audio, AudioVisual, Image, Multimedia, and Video) and

Table 21.1

Summary of the different MPEG-7-based multimedia ontologies

	Hunter [9]	DS-MIRF [10]	Rhizomik [8]	COMM
Foundations	ABC	None	None	DOLCE
Complexity	OWL-Full	OWL-DL	OWL-DL	OWL-DL
URL	metadata. net/mpeg7/	www.music.tuc.gr/ ontologies/MPEG703. zip	rhizomik.net/ ontologies/ mpeg7ontos	multimedia. semanticWeb.org/ COMM/
Coverage	MDS+Visual	MDS+CS	All	MDS+Visual
Applications	Digital libraries, eResearch	Digital libraries, eLearning	Digital rights management, e-business	Multimedia analysis and annotations

the decompositions from the MPEG-7 Multimedia Description Schemes (MDS) part. The descriptors for recording information about the production and creation, usage, structure, and the media features are also defined. The ontology can be viewed in Protégé (<http://protege.stanford.edu/>) and has been validated using the WonderWeb OWL Validator (<http://www.mygrid.org.uk/OWL/Validator>). This ontology has usually been applied to describe the decomposition of images and their visual descriptors for use in larger semantic frameworks. Harmonizing through an upper ontology, such as ABC, enables queries for abstract concepts such as subclasses of *events* or *agents* to return media objects or segments of media objects. While the ontology has most often been applied in conjunction with the ABC upper model, it is independent of that ontology and can also be harmonized with other upper ontologies such as SUMO [15] or DOLCE [16].


In 2004, Tsinaraki et al. proposed the DS-MIRF ontology that fully captures in OWL DL the semantics of the MPEG-7 MDS and the Classification Schemes. The ontology can be visualized with GraphOnto or Protégé and has been validated and classified with the WonderWeb OWL Validator. The ontology has been integrated with OWL domain ontologies for soccer and Formula 1 in order to demonstrate how domain knowledge can be systematically integrated in the general-purpose constructs of MPEG-7. This ontological infrastructure has been utilized in several applications, including audiovisual digital libraries and eLearning. The DS-MIRF ontology has been conceptualized manually, according to the methodology outlined in [10]. The XML Schema simple datatypes defined in MPEG-7 are stored in a separate XML Schema to be imported in the DS-MIRF ontology. The naming of the XML elements are generally kept in the `rdf:IDs` of the corresponding OWL entities, except when two different XML Schema constructs have the same names. The mapping between the original names of the MPEG-7 descriptors and the `rdf:IDs` of the corresponding OWL entities is represented in an OWL DL mapping ontology. Therefore, this ontology will represent, for example, that the `Name` element of the MPEG-7 type `TermUseType` is represented by the `TermName` object property, while the `Name` element of the MPEG-7 type `PlaceType` is represented by the `Name` object property in the DS-MIRF ontology. The mapping ontology also captures the semantics of the XML Schemas that cannot be mapped to OWL constructs such as the sequence element order or the default values of the attributes. Hence, it is possible to return to an original MPEG-7 description from the RDF metadata using this mapping ontology. This process has been partially implemented in GraphOnto [17], for the OWL entities that represent the `SemanticBaseType` and its descendants. The generalization of this approach has led to the development of a transformation model for capturing the semantics of any XML Schema in an OWL DL ontology [18]. The original XML Schema is converted into a main OWL DL ontology, while an OWL DL mapping ontology keeps track of the constructs mapped in order to allow circular conversions.

In 2005, Garcia and Celma presented the Rhizomik approach that consists of mapping XML Schema constructs to OWL constructs, following a generic XML Schema to OWL together with an XML to RDF conversion [8]. Applied to the MPEG-7 schemas, the resulting ontology covers the whole standard as well as the Classification Schemes and TV-Anytime (<http://tech.ebu.ch/tvanytime>). It can be visualized with Protégé or Swoop

(<http://code.google.com/p/swoop>) and has been validated and classified using the WonderWeb OWL Validator and Pellet. The Rhizomik ontology was originally expressed in OWL Full, since 23 properties must be modeled using an `rdf:Property` because they have both a datatype and object-type range, that is, the corresponding elements are both defined as containers of complex types and simple types. An OWL DL version of the ontology has been produced, solving this problem by creating two different properties (`owl:DatatypeProperty` and `owl:ObjectProperty`) for each of them. This change is also incorporated into the XML2RDF step in order to map the affected input XML elements to the appropriate OWL property (object or data type), depending on the kind of content of the input XML element. The main contribution of this approach is that it benefits from the great amount of metadata that has been already produced by the XML community. Moreover, it allows the automatic mapping of input XML Schemas to OWL ontologies and XML data based on them to RDF metadata following the resulting ontologies. This approach has been used with other large XML Schemas in the Digital Rights Management domain, such as MPEG-21 and ODRL [19], and in the eBusiness domain [20].

In 2007, Arndt et al. have proposed COMM, the *Core Ontology of Multimedia*, for annotation. Based on early work [21, 22], COMM has been designed manually by reengineering completely MPEG-7 according to the intended semantics of the written standard. The foundational ontology DOLCE serves as the basis of COMM. More precisely, the Description and Situation (D&S) and Ontology of Information Objects (OIO) patterns are extended into various multimedia patterns that formalize the MPEG-7 concepts. The use of an upper-level ontology provides a domain-independent vocabulary that explicitly includes formal definitions of foundational categories, such as processes or physical objects, and eases the linkage of domain-specific ontologies because of the definition of top-level concepts. COMM covers the most important part of MPEG-7 that is commonly used for describing the structure and the content of multimedia documents. Current investigations show that parts of MPEG-7 that have not yet been considered (e.g., navigation and access) can be formalized analogously to the other descriptors through the definition of other multimedia patterns. COMM is an OWL DL ontology that can be viewed using Protégé. Its consistency has been validated using Fact++-v1.1.5. Other reasoners failed to classify it due to the enormous amount of DL axioms that are present in DOLCE. The presented OWL DL version of the core module is just an approximation of the intended semantics of COMM since the use of OWL 1.1 (e.g., qualified cardinality restrictions for number restrictions of MPEG-7 low-level descriptors) and even more expressive logic formalisms are required for capturing its complete semantics.

To compare the four MPEG-7-based ontologies described above, consider a task to annotate the famous “Big Three” picture, taken at the Yalta (Crimea) Conference, showing the heads of government of the USA, the UK, and the Soviet Union during World War II. The description could be obtained either manually or automatically from an annotation tool. It could also be the result of an automatic conversion from an MPEG-7 description. The annotation should contain the media identification and locator, define the still region

SR1 of the image, and provide the semantics of the region using <http://en.wikipedia.org/wiki/Churchill> for identifying the resource Winston Churchill.  Figure 21.1 depicts the RDF descriptions generated for these four ontologies.

The link between a multimedia ontology and any domain ontologies is crucial. In the example, a more complete description could include information about “Churchill” (a person, a British Prime Minister, etc.) and about the event. In addition, details about the provenance of the image (e.g., date taken, photographer, camera used) could also be linked to complete the description. The statements contained in the descriptions above, in conjunction with any of the four underlying ontologies presented in this paper, can then be used to answer queries such as “*find all images depicting Churchill*” or “*find all media depicting British Prime Ministers*.” Furthermore, subjective queries such as “*find images with a ‘bright’ segment in them*,” where “bright” is defined as `mpeg7:DominantColor` greater than `rgb(220,220,220)`, are also possible.

Hunter’s MPEG-7 and COMM ontologies both use an upper ontology approach to relate with other ontologies (ABC and DOLCE). Hunter’s ontology uses either semantic relations from MPEG-7, such as `depicts`, or defines external properties that use an MPEG-7 class, such as `mpeg7:Multimedia`, as the domain or range. In COMM, the link with existing vocabularies is made within a specific pattern: the *Semantic Annotation Pattern*, reifying the DOLCE Ontology of Information Object (OIO) pattern. Consequently, any domain-specific ontology goes under the `dolce:Particular` or `owl:Thing` class. The DS-MIRF ontology integrates domain knowledge by subclassing one of the MPEG-7 `SemanticBaseType`: places, events, agents, etc. Furthermore, it fully captures the semantics of the various MPEG-7 relationships represented as instances of the `RelationType`. According to the standard, the value of these properties must come from some particular classification schemes: `RelationBaseCS`, `TemporalRelationCS`, `SpatialRelationCS`, `GraphRelationCS`, and `SemanticRelationCS`. A typed relationship ontology extending DS-MIRF has been defined for capturing all these relationships.

An important modeling decision for each of the four ontologies is how much they are tied to the MPEG-7 XML Schema. These decisions impact upon the ability of the ontology to support descriptions generated automatically and directly from MPEG-7 XML output and on the complexity of the resulting RDF. Therefore, the modeling choices also affect the scalability of the systems using these ontologies and their ability to handle large media datasets and cope with reasoning over very large quantities of triples. Both the DS-MIRF and the Rhizomik ontologies are based on a systematic one-to-one mapping from the MPEG-7 descriptors to equivalent OWL entities. For the DS-MIRF ontology, the mapping has been carried out manually, while for the Rhizomik ontology, it has been automated using an XSL transformation and it is complemented with an XML to RDF mapping. This has been a key motivator for the Rhizomik ontology and the ReDeFer tool where the objective is to provide an intermediate step before going to a more complete multimedia ontology, such as COMM. The advantage of the one-to-one mapping is that the transformation of the RDF descriptions back to MPEG-7 descriptions may be automated later on. In addition, this approach enables the exploitation of legacy data and allows existing

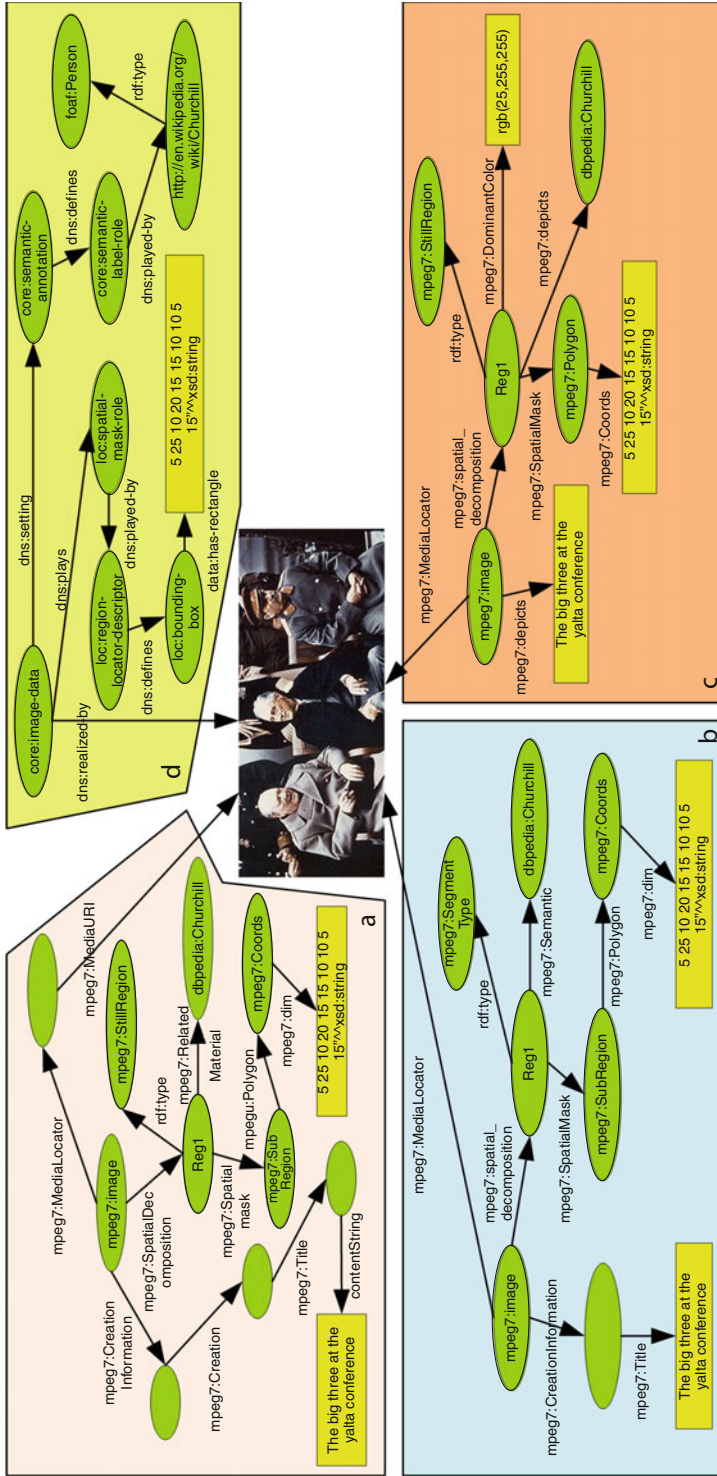


Fig. 21.1

An image described according to an MPEG-7 ontology. (a) Rhizomik approach. (b) DS-MIRF approach. (c) Hunter approach. (d) COMM approach. The image is a visual representation of the resource identified by http://en.wikipedia.org/wiki/Image:Yalta_Conference.jpg

tools that output MPEG-7 descriptions to be integrated into a semantic framework. The main drawback of this approach is that it does not guarantee that the intended semantics of MPEG-7 is fully captured and formalized. On the contrary, the syntactic interoperability and conceptual ambiguity problems such as the various ways of expressing a semantic annotation remain.

The COMM ontology avoids doing a one-to-one mapping for solving these ambiguities that come from the XML Schemas, while an MPEG-7-to-COMM converter is still available for reusing legacy metadata. A direct translation from an MPEG-7 XML description using Hunter's ontology is possible. However, in practice, the multimedia semantics captured by the ontology have instead been used to link with domain semantics. Therefore, rather than translating MPEG-7 XML descriptions into RDF, this ontology has been used to define semantic statements about a media object and to relate these statements to the domain semantics. This results in a smaller number of triples.

The MPEG-7-based ontologies discussed here aim to provide richer semantics and better frameworks for multimedia description and exchange than can be addressed by current standards. For further reading, the interested reader can also refer to [116] that surveys the state of the art of MPEG-7-based ontologies. Related efforts to develop multimedia ontologies include the following: the Visual Descriptor Ontology (VDO) [23] is based on the MPEG-7 Visual part and used for image and video analysis; [24] have proposed a visual ontology by extending WordNet with multimedia semantics from Hunter's ontology, specifically for use within the museums and art domain; [25] developed an MPEG-7-based ontology and applied it to annotating football (soccer) videos; similar to the approach used in Hunter's ontology and in COMM, this ontology uses the decomposition and visual components of MPEG-7 and captures high-level domain semantics in domain-specific ontologies.

Toward a Standardized Ontology for Media Resources

The Ontology for Media Resources currently being specified in W3C is a core vocabulary that covers basic metadata properties to describe media resources (see www.w3.org/TR/mediaont-10/). The goal of this ontology is to address the interoperability problem by providing a common set of properties defining the basic metadata needed for media resources and the semantic links between their values in different existing vocabularies. The ontology can be used to attach different types of metadata to the media, such as the duration, the target audience, the copyright, the genre, and the rating. Media fragments can also be defined in order to have a smaller granularity and attach keywords or formal annotations to parts of the video. The ontology will also be accompanied by an API that provides uniform access to all elements defined by the ontology.

The purpose of the mappings defined in the ontology is to enable different applications to share and reuse metadata represented in heterogeneous metadata formats. For example, `creator` is a common property that is supported in many metadata formats. Therefore, it is defined as one of the properties in the core vocabulary of the ontology for media resources and aligned with other vocabularies. Ideally, the mappings defined in the ontology should be used to reconcile the semantics of a term defined in a particular

```
<http://data.linkedevents.org/media/4303994975>
  a ma:Image;
  dc:title "Radiohead / Thom Yorke";
  ma:locator <http://farm3.static.flickr.com/2726/4303994975_74302c45b5_o.png>;
  ma:createDate "2010-01-25T12:27:21"^^xsd:dateTime;
  ma:frameWidth "1280"^^xsd:integer;
  ma:frameHeight "720"^^xsd:integer;
  ma:keyword "colin";
  ma:keyword "radiohead".
```

■ Fig. 21.2

MediaOntology annotation (Courtesy Raphaël Troncy, from data.linkedevents.org)

schema. However, this cannot be easily achieved, due to the many differences in the semantics that are associated with each property in the mapped vocabularies. For example, the property `dc:creator` from Dublin Core and the property `exif:Artist` defined in EXIF are both aligned to the property `ma:creator`. However, the extension of the property in the EXIF vocabulary (i.e., the set of values that the property can have) is more specific than the corresponding set of values that this property can have in Dublin Core. Therefore, mapping back and forth between properties from different schemas, using this ontology as a reference, will induce a certain loss in semantics. The axioms representing the mappings are defined as an exact, broader, or narrower mapping between two properties (► Fig. 21.2).

21.2.2 Semantic Web–Based Multimedia Annotation Tools

As already sketched by the aforementioned, multimedia annotations come in a multilayered, intertwined fashion, encompassing among others descriptions about the conveyed subject matter (e.g., a train arriving at the station in a rainy day), content structure (e.g., the specific image region depicting the train or the part of the video capturing the train as it approaches), visual features (e.g., the values of the color descriptors corresponding to the rainy sky image parts), administrative information (e.g., ownership and editing rights), and so forth. Different aspects pertain depending on the annotation needs and the particular application context addressed each time, as illustrated in the description of state-of-the-art ontology-based annotation tools that follows.

SWAD (<http://swordfish.rdfWeb.org/discovery/2004/03/w3photo/annotate.html>), though no longer maintained, constitutes one of the first Semantic Web–based implementations addressing the manual annotation of images. Through a Web-based interface, it allows the user to insert descriptions regarding who or what is depicted in an image (person, object, and event), when and where it was taken, and additional creation and licensing information. Annotations are exported in RDF. Despite the very early stage of Semantic Web technologies uptake, SWAD used an impressive number of

RDF vocabularies including FOAF, the Dublin Core element set, RDFiCalendar (<http://www.w3.org/2002/12/cal/>), as well as an experimental, at that time, namespace for WordNet.

PhotoStuff (<http://www.mindswap.org/2003/PhotoStuff/>) is a platform-independent ontology-based annotation tool that allows users to describe the contents of an image with respect to ontology concepts, as well as administrative information about the image such as the creation date [26]. Multiple RDF(S) or OWL ontologies can be simultaneously loaded, facilitating the user in the creation of annotations distributed across many ontologies. Classes from the ontologies can be associated to the entire image or specific regions using one of the available drawing tools (circle, rectangle, and polygon), while the necessary region, localization, and so forth, definitions are provided by a built-in image-region ontology. Annotations referring to relations can also be created by connecting concept annotation instances that have been already identified in an image using properties from the uploaded ontologies. PhotoStuff also takes advantage of existing metadata embedded in image files (such as EXIF) by extracting and encoding such information in RDF/XML. The property `depicts` from FOAF and its inverse (`depiction`) are used to link image (region) instances with domain ontologies concept instances. Finally, PhotoStuff supports browsing, searching, and managing digital image annotations through a loosely coupled connection with a Semantic Web portal (➤ [Fig. 21.3](#)).

AktiveMedia (<http://www.dcs.shef.ac.uk/~ajay/html/cresearch.html>), developed within the AKT (<http://www.aktors.org/akt/>) and X-Media (<http://www.x-media-project.org/>) projects, is an ontology-based cross-media annotation system addressing text and image assets [27]. In the image annotation mode, AktiveMedia supports content markup with respect to multiple domain-specific ontologies. Unlike PhotoStuff, only a single ontology is displayed each time in the ontology browser. The user-created annotations may refer to an image or a region level (using the provided rectangular and circular drawing facilities), and a simple built-in schema is used to capture localization information. To describe an image as whole, AktiveMedia provides three free text fields, namely, title, content, and comment. Using the text annotation mode, the user-entered descriptions can be subsequently annotated with respect to an ontology. The supported ontology languages include RDFS and OWL, as well as older Semantic Web languages such as DAML and DAML-ONT, and RDF are used for the export of the generated annotations. An interesting feature of AktiveMedia, though not directly related to the task of image annotation, is its ability to learn during textual annotation mode, so that suggestions can be subsequently made to the user (➤ [Fig. 21.4](#)).

Following a different rationale, M-Ontomat-Annotizer, developed within the aceMedia project (<http://www.acedmedia.org/aceMedia>), extends the typical media annotation by enabling in addition the formal representation of low-level visual features and their linking with concepts from a domain ontology [28]. In order to formalize the linking of domain concepts with visual descriptors, M-Ontomat-Annotizer employs the Visual Annotation Ontology (VAO) and the Visual Descriptor Ontology (VDO) [29], both hidden from the user. The VAO serves as a meta-ontology allowing one to model domain-specific instances as prototype instances and to link them to respective descriptor

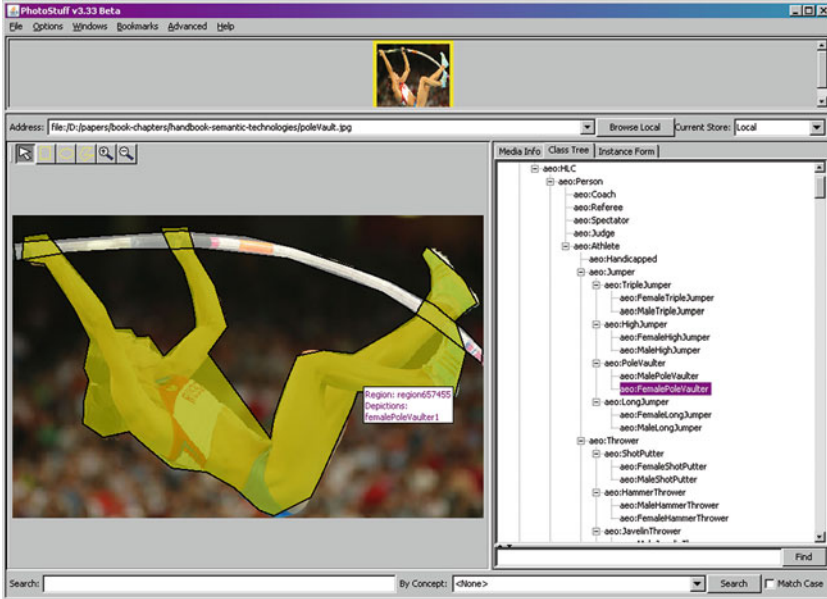


Fig. 21.3
Annotation screenshot using PhotoStuff

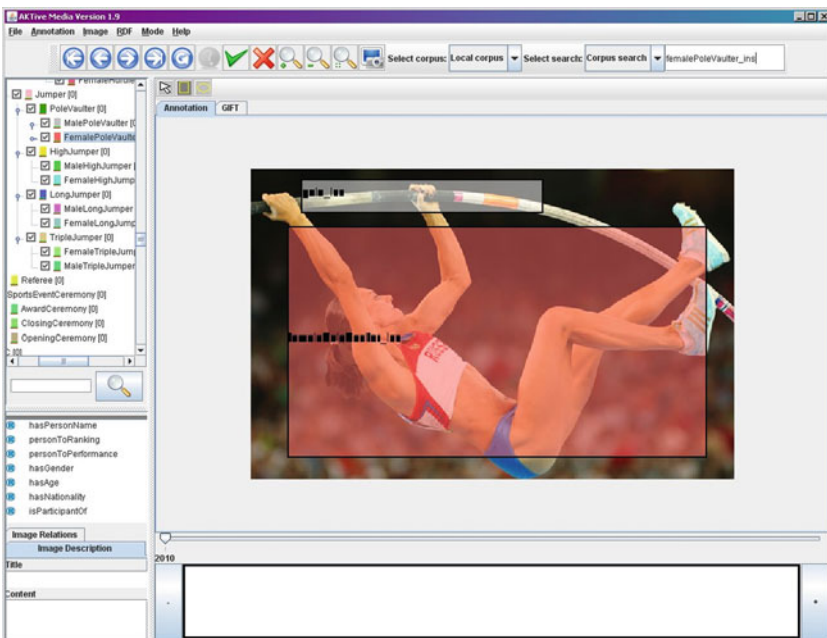
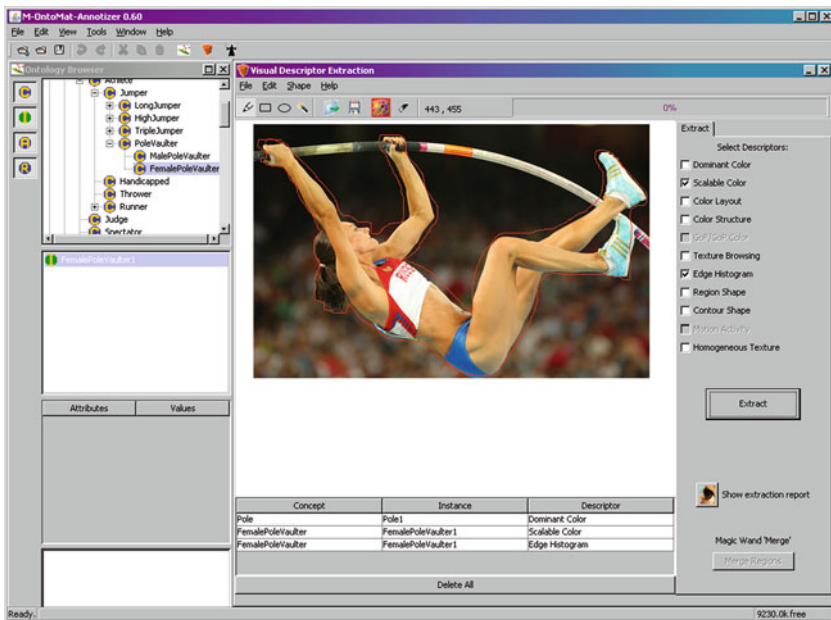


Fig. 21.4
Annotation screenshot using ActiveMedia

instances through the *hasDescriptor* property. The domain-specific instances, and by analogy the extracted descriptor instances, may refer to a specific region or to the entire image. For the identification of a specific region, the user may either make use of the automatic segmentation functionality provided by the M-Ontomat-Annotizer or use one of the manually drawing tools, namely, the predefined shapes (rectangle and ellipse), free hand, and magic wand. The supported input ontology languages are RDFS and DAML. In a subsequent release within the K-Space project (<http://kspace.qmul.net>), M-Ontomat 2.0 (<http://mklab.iti.gr/m-onto2>) provides support for descriptive and structural annotations in the typical semantic search and retrieval sense (• Fig. 21.5).

The K-Space Annotation Tool (KAT) (<https://launchpad.net/kat>) is an ontology-based framework developed within the K-Space project for the semiautomatic annotation of multimedia content [30]. Its core provides the infrastructure of an API and set of services, including configuration and access to Sesame and Sesame2 repositories that enable users to implement in a plug-in-based fashion relevant functionalities such as visualization, editing, and manipulation of semantic content annotations. The model and storage layer of KAT is based on the Core Ontology of Multimedia (COMM) [31] and the MultiMedia Metadata Ontology (M3O), a subsequent extension that addresses also the annotation of rich multimedia presentations [32]. In the current release, concepts from an ontology can be used to mark up images and respective regions that are localized manually, using either the rectangle or the polygon drawing tools. Decomposition and localization information



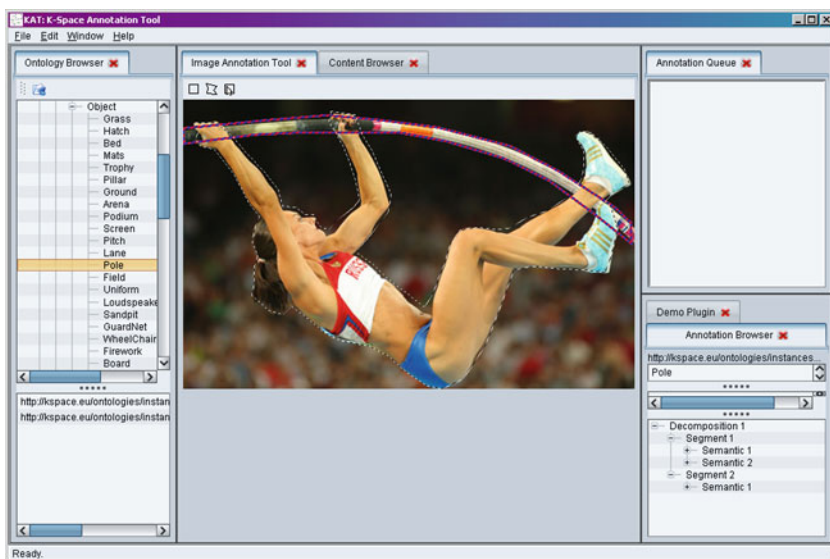
• Fig. 21.5

Annotation screenshot using M-Ontomat-Annotizer

is represented based on the respective COMM decomposition pattern. KAT's flexible architecture and COMM-based annotation model render it media independent, allowing support for additional content types as long as respective media management functionalities (e.g., video player) are implemented. Furthermore, the COMM-based annotation model makes it quite straightforward to extend annotation level so as to include additional dimensions, such as low-level visual features for example, again as long as appropriate feature extraction plug-ins are available (► Fig. 21.6).

The Video and Image Annotation (VIA) tool, developed within the BOEMIE project (<http://www.boemie.org>), provides a looser notion of ontology-based media markup. Specifically, it supports the annotation of image and video assets using concepts from an ontology, while allowing also for free text descriptions. Users may also add administrative descriptions, including information about the creator of the annotations, the date of the annotation creation, etc., based on a simple built-in schema. Image (and video frame) annotation may address the entire image (video frame) or specific regions; in the case of image annotation, the user can also select to extract MPEG-7 visual descriptors to enhance annotations with low-level information. The localization of regions is performed either semiautomatically, providing to the user a segmented image and allowing him or her to correct it by region merging, or manually, using one of the drawing functionalities, namely, free hand, polygon, circle, or rectangle.

Video annotation may refer to the entire video asset, video segments, moving regions, frames, or still regions within a frame. It can be performed either in a successive frame-by-frame fashion or in real time, where the user follows the movement of an object while the

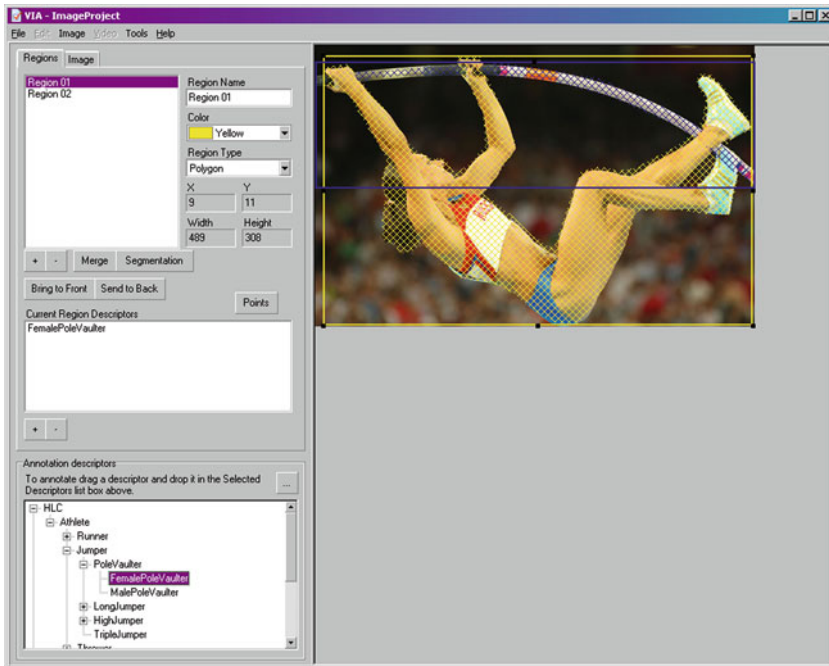


► Fig. 21.6

Annotation screenshot using KAT

video is playing, by dragging its bounding box. The annotations performed using VIA can be saved as annotation projects, so that the original video, the imported ontologies, and the annotations can be retrieved and updated at a later time. The produced metadata are exported following a custom format, either in XML or in a more human-readable textual form (► Fig. 21.7).

Following a different approach, a number of media annotation tools have been developed based on MPEG-7 and customary multimedia description vocabularies. This line of thought is particularly evident in the case of video annotation, where Semantic Web–based technologies have hardly been employed; KAT is an exception, though currently it provides just the infrastructure and not an implementation, while VIA, though allowing the use of a subject matter ontology for video annotation, uses a proprietary format for encoding the generated metadata. Prominent examples of non-Semantic Web compliant video annotation tools include IBM’s VideoAnnEx, Anvil, the Semantic Video Annotation Suite, Ontolog, Elan, etc. (despite some names, none of these tools produces metadata based on an ontology). For a complete list of relevant tools and resources for the annotation of media content, the interested reader is referred to the Tools and Resources page of W3C Multimedia Semantics Incubator Group (http://www.w3.org/2005/Incubator/mmsem/wiki/Tools_and_Resources).



► Fig. 21.7

Annotation screenshot using VIA

Besides the uneven uptake of Semantic Web technologies in the annotation of video assets compared to images, the aforementioned tools lead to a number of considerations. A critical one relates to the interoperability of the generated annotation metadata. The different ontology schemas used by the individual tools to represent media-related descriptions and the respective modeling approaches to the linking with domain ontologies hamper the sharing and reuse of annotations across different applications. The situation is further aggravated, as many tools choose to follow proprietary schemas. This discrepancy between the available theoretical results and their uptake in practical applications brings forth once again the trade-off between the complexity of the proposed formal representation models and the effective fulfillment of real-world needs. It is also quite interesting to note that many tools do not allow users to edit previously created annotations at a later point, treating annotation as a onetime affair.

The ambiguity that characterizes the relation and interlinking between annotations generated by different tools induces a subsequent vagueness when assessing the appropriateness of each tool for a given application. For semantic search and retrieval, applications that address content at the level of perceived meaning, possible selection criteria may be the expressivity level (are ontology classes adequate or relation descriptions are also needed?) or the granularity of the annotations (depth of spatial or temporal content decomposition). For applications that encompass multimedia analysis aspects too, tools that support descriptions at the level of low-level features provide the means to capture and share this additionally required information.

Concluding, although the main focus of semantic multimedia research continues to be the automatic extraction of multimedia content annotations, the ability to effectively generate manually or semiautomatically annotations remains a crucial pursuit. Despite the strenuous research efforts and successful results in sporadic application domains, the automatic extraction of multimedia semantics is still at a very naive level compared to practical user needs. Moreover, manual content annotations contribute actively in semantic multimedia research serving as ground truth data for evaluation purposes and as training data to support knowledge acquisition and learning tasks.

21.2.3 Semantic Multimedia Analysis

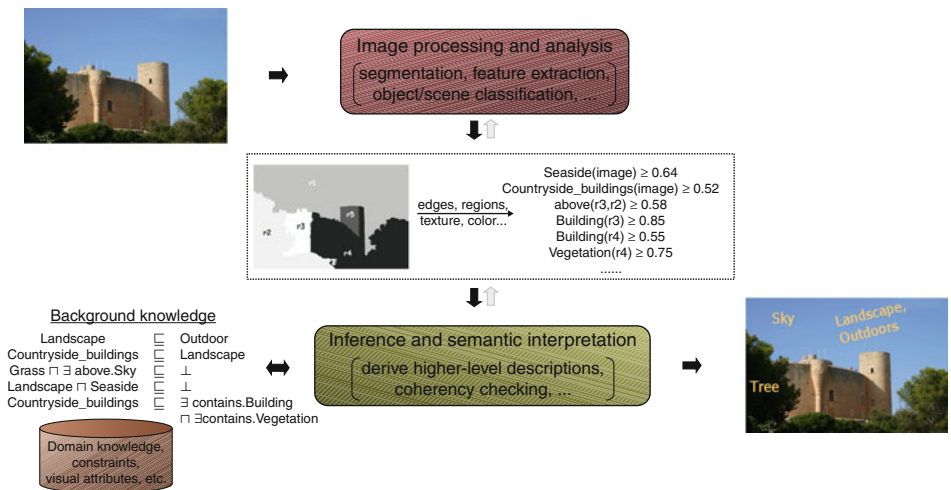
Automated multimedia content understanding has strained researchers for years in the painstaking quest to confront the so-called semantic gap challenge, namely, the lack of correspondence between the low-level content descriptions that can be automatically derived and the semantic interpretation a human would attribute [33].

Since its early days, research in content understanding has been intertwined with the use of structured, prior knowledge in the pursuit of endowing computational systems with the notion of informed (in terms of background knowledge-driven) interpretation. This interrelation has rendered knowledge representation and reasoning as integral elements of the undertaken investigations.

In the 1980s and early 1990s, semantic networks and frames were widely used to model the relevant domain knowledge and support object recognition and scene interpretation tasks, while formal approaches, based on first-order logic, were significantly sparser [34, 35]. The lack of consensual semantics and the customized reasoning algorithms, hindered the sharing and reuse of knowledge across systems, and often led to disparate conclusions for the same problem. Furthermore, the underlying assumption that all aspects involved in semantic content analysis should be explicitly modeled had resulted in extremely elaborate conceptualizations and decision-making strategies. All these led gradually to a period of rather receding interest in the use of explicit knowledge, a tendency further corroborated by the momentum that statistical inference approaches have gained as generic tools for semantic image and object classification. Soon though, the limitations of relying solely on learning using perceptual information and similarity-based associations became apparent, reviving interest into the role of knowledge and reasoning in multimedia content understanding [36, 37]. Following the Semantic Web initiative, ontology [115] and Description Logic (DL) languages [116] became the prevalent formalisms for capturing and representing knowledge, establishing the current literature.

The following considers the use and role of Semantic Web technologies in the current state of the art in semantic multimedia analysis and understanding, and concludes with a brief discussion on open issues and challenges for future directions. As space constraints have enforced several simplifications and omissions, the interested reader is referred to the provided related resources for a thorough treatment of the topics addressed.

► *Figure 21.8* depicts a typical example framework for semantic image analysis deploying Semantic Web–based technologies. The interpretation process starts with an image processing stage, where the extraction of relevant low- and intermediate-level



■ **Fig. 21.8**
Semantic image analysis architecture

representations pertaining to perceptual features, such as color and texture, takes place. This is often carried out with the joint application of spatial (temporal) segmentation. The extracted representations may then be used directly as input facts (assertions) in the inference process, or may undergo further processing to acquire descriptions of higher abstraction (e.g., concept classifiers for the detection of primitive objects, as in the illustrated example), before inference is eventually invoked. The knowledge base encodes logical associations and constraints necessary to admit valid content interpretations, as well as appropriate abstraction layers so as to link and map information derived from perceptual evidence to object and scene interpretations. A possible model for a building thus, may span multiple levels of abstraction, starting with constituent parts such as windows and doors, include corresponding spatial and geometric characteristics, and reach down to edge and spatial adjacency definitions at the level of image pixels.

Knowledge modeling and inference lie at the core of the interpretation process and being intertwined with the configuration of content interpretation, they comprise the chief dimensions of differentiation in the current state of the art. For example, approaches that realize interpretation as a stepwise transition from low- to high-level content representations, place naturally large emphasis on the modeling and linking of media-related knowledge to domain-specific knowledge. Approaches that tackle instead the configuration of content interpretation in a formally accountable way tend to concentrate on the implications and adaptation requirements imposed on inferencing, and usually abstract lower-level representations. Additional differences can be traced to the intrinsic ambiguity involved in content interpretation that gives rise to diverse imprecision handling methodologies, to conceptual modeling requirements that affect the granularity and types of knowledge considered, to the interplay between knowledge, inference, and analysis, and so forth, to name but a few.

In the following, consider the following representative examples from the current state of art, outlining the main characteristics, weaknesses, and insights, starting with approaches that consider the formal representation of media-related knowledge, and which can be further classified into those adhering to standardized definitions, such as MPEG-7, and those following proprietary ones. In [38] domain experts define, through a graphical interface, rules that map particular combinations of low-level visual features (color, texture, shape, and size) to high-level semantic concepts defined in the domain ontology. These rules are subsequently applied in order to infer descriptions of the form “still region ri depicts cj ,” where cj is an instance of a domain concept [39, 40]. In [23], MPEG-7 compliant visual descriptions (color, texture, shape) are used to enrich domain concepts serving as prototypical visual instantiations. These enriched domain representations are subsequently used as prior knowledge to train statistical classifiers for the automated detection of the addressed semantic concepts. The semantic link between these prototypical instances and the respective domain concepts is established through the M-Ontomat-Annotizer graphical annotation tool [41] making use of the Multimedia Structure ontology, the Visual Descriptor ontology, and the Visual Annotation ontology [29].

The MPEG-7 compliant media-related knowledge representations render the aforementioned approaches particularly appealing in terms of reusing and sharing the knowledge involved. The extracted low-level features can be interchanged straightforwardly

between different applications, saving significant time and effort that would be required for their recalculation. Similarly, the inference rules defined by domain experts and enriched with visual attributes can be shared across applications enabling them to more effectively communicate human domain knowledge, where relevant.

Addressing similar considerations, a visual ontology-guided approach to object recognition is presented in [42], this time adopting a proprietary approach to the definition of the media-related descriptions. Domain experts populate the knowledge base using the color, texture, and spatial concepts provided by the visual ontology to describe semantic objects of the domain. An image processing ontology is used to formally encode notions pertaining to the image processing level, including entities such as edge and region, and features, such as color histograms to numerically characterize visual properties. The linking of the visual ontology descriptions with the image processing ontology descriptions representations can be either manually predefined or learned following a mixed bottom-up and top-down methodology [43, 44]. Another processing-related ontology is presented in [45], this time accounting for the algorithmic aspects of the analysis process for the detection of semantic objects in video sequences. A domain-specific ontology provides object descriptions extended with low-level and qualitative visual knowledge, and a set of rules determines the sequence of steps required to detect particular semantic objects using the definitions provided by the analysis ontology. In [116], clusters of visually similar content, computed based on low-level features, are used as concept definitions to enrich (via subclass relations) the linguistic terms comprising the domain ontology.

Besides differences in the modeling and engineering of the background knowledge, the aforementioned approaches share a common underlying assumption, modeling interpretation as straight bottom-up deduction by augmenting the initial perceptual facts through inference upon the available background knowledge. Although this assumption may be true in given applications, the incompleteness, ambiguity, and complexity that characterize the task of content interpretation, in general, render such a view of limited applicability. To meet the challenge of selecting among plausible alternatives while constructing an interpretation, a number of approaches have investigated more closely the requirements imposed on inferencing.

In a series of works [46–48], Description Logics are examined for high-level scene interpretation based on the notion of *aggregates*, that is, concepts that consist of multiple parts that are constrained with respect to particular spatial (or temporal) relations. High-level concepts are linked to corresponding *view-concepts*, which realize the grounding with low-level processing evidence and initiate the (partial) instantiation of aggregates. The interpretation process is modeled as a recursive search in the space of alternative interpretations exploiting the logical structure of the aggregates in a mixed bottom-up and top-down fashion. The existence of multiple models though leaves open a great degree of freedom for choosing which of the alternatives to examine first each time. In [49], a middle layer serves as mediator, attempting to match hypotheses from the high-level interpretation layer to the available evidence; if a hypothesis is neither confirmed nor refuted, low-level image processing is invoked again with accordingly updated parameters. Work toward a probabilistic model for handling dependencies in compositional aggregate

hierarchies is sketched in [50], with the purpose of providing preference measures to guide the interpretation steps.

In [51], the media interpretation configuration described above is extended to formalize interpretation as abduction over Description Logic ABoxes. The set φ of input assertions that are provided by image analysis are split into bona fide assertions φ_1 that are considered true by default, and bona fiat ones φ_2 that need to be explained. Interpretation then is formalized as the abductive problem of finding the explanations α such that $\Sigma, \varphi_1, \alpha \models \varphi_2$, holds. In the presented experimental setting, φ_2 corresponds to the set of spatial relationships assertions. Such division, however, is arbitrary and cannot be justified formally; similar considerations hold for the definition of the backward-chaining rules used to implement the abductive reasoning. Preference over the possible explanations is determined in terms of the number of (new) individuals that need to be hypothesized (as part of α) and the number of φ_2 assertions that get explained.

Advancing from purely deductive configurations to include nonstandard forms of inference marks a significant turn, given the ill-defined transition from perceptual representations into semantic descriptions. Automatic segmentation hardly ever results in (semantically) meaningful partitions, while it is practically impossible to acquire unique and reliable mappings between perceptual appearances and semantic notions. In such a setting, the ability to cope with incompleteness and ambiguity is crucial, and abductive reasoning, providing inference to the best explanation, presents an appealing direction for future research.

However, extensions of this kind are not sufficient alone. The extraction of media semantics at the level of objects, events, and scenes encompasses intrinsically a large amount of imprecision. It permeates the extraction of features, the identification of shapes, matching textures, colors, etc., and distills the translation from perceptual to symbolic representations addressed by image analysis. The latter may express either uncertainty, thus representing degrees of belief and plausibility, or vagueness, expressing conformity through degrees of membership [52]. Yet, the majority of the literature tends to treat the descriptions upon which inference is performed as crisp facts (binary propositions), ignoring the probability or vagueness information captured in the accompanying confidence degrees.

The need to accommodate for vagueness has been acknowledged early on. In [53], a preliminary investigation into the use of Description Logics for object recognition is reported, outlining the limitations involved with exact recognition; in a subsequent investigation, the proposed framework has been extended with approximate reasoning to assess composite shape matching and subsumption [54]. In [55], a fuzzy DLs-based reasoning framework is proposed to integrate, possibly complementary, overlapping, and/or conflicting classifications at object and scene level, into a semantically coherent final interpretation. The input classifications, obtained by means of statistical learning, are modeled as fuzzy assertions, and a three-step procedure is followed in order to determine the set of plausible interpretations, resolve inconsistencies by tracking the assertions and axioms triggering them, and further enrich the interpretations by making explicit missing descriptions [56]. Other approaches building on fuzzy semantics include [57], where

fuzzy DLs are used to merge over-segmented regions and to accordingly update the degrees of classifications associated to the regions, and [58], where a fuzzy ontology capturing spatial relations for image interpretation is presented.

Similar to fuzzy extensions [59, 60], probabilistic extensions to logical inference [61, 62] have been investigated in media interpretation approaches, though significantly sparser. In [63], an appropriately defined ontology, which links visually extracted descriptors with domain entities and semantic constraints, guides the design of the Bayesian network used to perform probabilistic inference over automatically extracted video descriptions. In [64], commonsense knowledge, encoded in the form of first-order logic production rules, is used to deduce the topology of a Markov Logic Network [65] and semantically analyze parking lot videos. The use of the Markov Logic Network makes it possible to formulate the uncertainty involved in the detection of objects and movements, as well as the statistical ambiguity characterizing part of the domain knowledge; while in [118], bilattice theory, which orders knowledge along two axes that represent the degree of truth and the degree of belief, respectively [124], is explored as the means to handle and reason under imprecision in human detection applications. In [119], a reasoning framework that combines Semantic Web technologies with rule-based and causality-based reasoning is investigated, while highlighting challenges with respect to inconsistency and uncertainty handling. Finally, it is worth mentioning two initiatives that culminate the findings of a series of workshops toward an ontology framework for representing video events. The Video Event Representation Language (VERL) models events in the form of changes of states, while the Video Event Markup Language (VEML) serves as a complementary annotation framework [66]. Though less rigorous than respective logic-based formalisms for representing actions and effects and temporal semantics (e.g., the Event Calculus [120]), such initiatives manifest a continuously increased awareness and interest in cross-disciplinary results and experiences.

The aforementioned work outlines an intriguing amalgam of valuable results and insightful observations. As illustrated by the current state of the art, formal knowledge representation and reasoning bring in a tremendous potential to inject semantics into the otherwise data-driven statistical learning and inferencing technologies used in media interpretation. Intrinsic traits challenge the typical deductive reasoning scheme much as the classical binary value semantics, demanding a profound investigation of the extensions and adaptations necessary to the currently available inference mechanisms. In this quest, the management of imprecision is crucial, especially with regard to the effective combination of probabilistic and fuzzy semantics under a formal, coherent framework: the distinction between probable and plausible interpretations is key both to forming and ranking alternative interpretations.

Supporting hybrid inference schemes that allow for imprecision though, is not sufficient on its own to handle the missing and incomplete descriptions obtained by means of typical media processing. Building on the classical logic paradigm, Semantic Web languages adopt the open-world assumption. Low-level representations serve as evidence that determine the set of possible interpretations and formal knowledge is expected to further restrict them into valid ones based on coherency and consistency

considerations; yet compositional semantics are hardly encountered in the existing literature. The investigated interpretation configurations implicitly espouse a closed world-view, focusing only on explicitly asserted facts, while poorly exploiting the supported open semantics in the involved knowledge modeling and engineering tasks [66]. Such requirements are intertwined with another critical challenge, namely, the transition from pipeline-like interpretation configurations, where successive steps of statistical and logical inference take place, to more interactive schemes that exploit jointly learning and logical inference. The existing approaches address fragmentarily and only partially the aforementioned considerations, paving an interesting roadmap for future research activities.

21.2.4 Semantics in Broadcasting

21.2.4.1 Metadata in Broadcasting from Its Origin

Although it was not called “metadata,” the audiovisual industry and the broadcasters in particular have been managing such content-related information for decades. Archives from where content has to be found and retrieved have been the place where the need for accurate documentation first arose.

Metadata is the modern IT equivalent of a label on a tape or film reel (title, short description) with potentially more structured machine-readable information (technical details, broadcast time, storage location). With a growing quantity of content being produced every year (thousands of hours of audio and video material), the business rationale behind well-documented metadata is more justified than ever: “if you can’t find it, it’s like you don’t have it, hence you must pay for it again!”

Although the first databases date back to the 1960s, their real expansion came with the democratization, ease of use, and the reasonable computing power of computers in the mid-1980s. Within the broadcasting community, the “early adopters” waited until the mid-1990s (already 15 years later) to measure the potential of metadata and information management in databases. Still, it is only recently that the role of metadata has been fully recognized.

In an analog world, the first broadcaster’s need for metadata was internal to recover all the information historically available on tags, cards, production forms, and a reference to a physical location of the media (e.g., tape, film, and disk on a shelf). Digitization has been the opportunity for generating and managing more data like restoration information (e.g., tools, methods, parameters, results). In a file-based production environment, metadata is vital: what is the format of the video or audio file? What editorial content does the file contain? Where is the resource within petabytes of distributed mass storage? The exchange of content (e.g., between the post-producer of an advertising spot and the broadcaster in charge of exploiting it) is also greatly facilitated by metadata to search material, publish information on programs available, and provide information on the file being provided.

However, although all the technical conditions are now met to develop effective metadata solutions for production, the cost of generating metadata remains a barrier and the next challenge is to develop tools to automatically extract or generate metadata. This includes speech-to-text recognition, face recognition, format detection, and content summarization (e.g., reduce a 40-min program into a 3-min clip made of representative key scenes and synchronized metadata).

Last but not least, the objective of broadcasters is to have their programs being easily accessed and seen across a variety of delivery media and platforms including traditional linear broadcast, but also Internet (live streaming, video-on-demand, catch-up TV), mobiles, and any hybrid combination like hybrid broadcast–broadband. In this rich and ubiquitous context, metadata is vital.

21.2.4.2 Metadata Standardization in Broadcasting

In this section, different metadata standards for production and distribution will be mentioned. Proprietary metadata solutions from MAM (Media Asset Management) solution providers or consumer electronics manufacturers (proposing competing program guides accessible through their respective products for an additional fee) are intentionally out of scope.

Different groups are working on broadcasting standards. The Advanced Media Workflow Association (AMWA) has a focus on metadata associated to container formats also carrying metadata (the Advanced Authoring and Media Exchange Formats, AAF, MXF). The European Broadcasting Union (EBU) is developing technical specifications related to all domains of broadcasting technology, including metadata. The Society of Motion Picture and Television Engineers (SMPTE) develops specification for audiovisual production. Harmonization is desired although difficult to achieve. But, it must be noted that several of the existing standards correspond to different needs or have only a regional impact.

Why Are Standards Necessary?

The “business-to-business exchange” application led to the necessity to propose a solution for interoperability, that is, using information understandable to the sending and receiving parties. It is critically needed in a broadcasting environment in which data, aggregated from different providers, have to be forwarded in a common format to receiving devices from different consumer electronics manufacturers. It remains true for hybrid broadcast–broadband services where data are also aggregated from different sources and represented in a common format, for example, for display on a portal page or for transmission to devices.

What Is Meant by Interoperability?

The *first level* of interoperability is the identification of a common set of structured attributes characterizing content with agreed names and detailed semantics. Some examples: DMS-1 has been defined by AMWA as a set of attributes to be associated to audiovisual material in MXF (Media Exchange Format) containers. RP210 is an SMPTE

dictionary of metadata attributes commonly met in television and radio production. The EBUCore is defined by EBU as a core set of metadata based on Dublin Core to facilitate the aggregation and exchange of metadata with audiovisual archive portals. ETSI TV-Anytime was developed to facilitate the use of personal video recorders through harmonized electronic program guide data. DVB Service Information is a minimum set of information related to programs and services, which is broadcast in DVB streams.

The *second* level of interoperability is the representation format, which defines how the structure of description attributes is being digitally serialized. Some examples of representation formats are:

- SMPTE KLV (Key, Length, Value)
- W3C XML, RDF/OWL N3, or Turtle
- JSON (JavaScript Object Notation)
- DVB SI (binary encoding of service information)

The *third* level of interoperability is the definition of delivery mechanisms (e.g., standardized by DVB in Europe, ARIB in Japan, or ATSC in the USA) over, for example, MPEG Transport Stream (MPEG-TS) or Internet Protocols (IP). This includes solutions adapted to the bandwidth of the different media such as data fragmentation and partial updates.

21.2.4.3 Using Ontologies: Metadata + Semantic

One motivation for broadcast metadata is to provide search and personalization functionality through access to richer information in order to facilitate faster queries and deliver results more relevant to users. This, in proportion with the large volumes of audiovisual material being produced, requires always more metadata augmented with more semantics. An important question to answer before designing an ontology and associated properties is “what is it that the implementer wants users to search for?”

Because of the close relation of the Semantic Web initiative to W3C, the use of semantic descriptions of audiovisual content was initially thought to have a de facto focus on distribution, that is, targeting access to content by the users. This is why work primarily started from TV-Anytime (a metadata format for describing electronic program guides and on-demand catalogs), which additionally proposes a consistent class model and embryonic identifier-based entity relationships. Further work showed the high potential value of also using semantic-based descriptions for metadata at the production stage and broadcaster archives.

21.2.4.4 A Semantic Representation of TV-Anytime in a Nutshell

The TV-Anytime specification has been developed within an open forum of broadcasters, manufacturers, operators, EPG providers, etc. It addresses linear broadcasting and online nonlinear services. Although it was first published by ETSI in the series of specifications

TS 102 822 in 2005, it also fits new content access concepts like catch-up TV and other mobile services.

TV-Anytime benefits from a solid class-oriented data model shown in [▶ Fig. 21.9](#).

[▶ Figure 21.9](#) shows different types of classes (e.g., `ProgramGroup`, `Programme`, `Person`), entity/class relationships (object properties in blue), and data properties (in red). Considering how TV-Anytime could be represented in a Semantic Web model:

- The set of classes forms the backbone of the model. All the classes (e.g., `ProgramGroup`, `Programme`, `Segment`) represented in [▶ Fig. 21.9](#) are properly identified (as they would likely be recorded in a database) and can easily be attributed a URI, which is a key eligibility criterion for a class in the Semantic Web. The fact that `CreditsItem` does not have an identifier is not essential in XML but it will be more critical in a semantic model to avoid using blank nodes through which a `Person` or `Organisation` class instance would be linked to `Programme` with the addition of the role data property. However, this means that credit items should be managed as an individually identified class in the broadcaster's database, which exists but is not necessarily common practice. Best practice would dictate that a forthcoming version of TV-Anytime contains an optional identifier per credit item.
- TV-Anytime relations such as `MemberOf`, `AggregationOf`, or `RelatedMaterial` are directly eligible to become object properties. It must be noted that several of these relations have their inverse also defined, which is another important feature in support of semantic models.
- IDRef relationships (from the XML Schema) are also implicit object properties for which better names can be found.
- XML implicit relationships like `ScheduleEvent`, an element member of the `Schedule` complex type would need to be associated with a proper identifier to become a `ScheduleEvent` class for which an object property such as `HasScheduleEvent` would be used to create an association with a class schedule.
- As far as data properties are concerned, the transformation is rather straightforward with the exception that reusable complex-type structures should be replaced by flat structures directly referring to a class. This again is to avoid blank nodes.

Starting with the transformation rules mentioned above, it becomes easy to transform the most significant part of the TV-Anytime model into an ontology written in RDF (Resource Description Framework) and OWL (Web Ontology Language). As an example, the statement “a TV Program has the title ‘Tonight’s Show’” could be expressed in RDF/OWL as shown in [▶ Fig. 21.10](#).

However, it is not necessarily optimal to work only in RDF/OWL. For example, cardinalities cannot be managed with the same flexibility as in XML. An option would therefore be to generate instances in the strongly validated XML environment, and to transform the results into an instance of the equivalent ontology as shown in [▶ Fig. 21.11](#).

The use of an instance template is attractive to users as it hides from them the complexity of the ontology. However, generating instance templates for complex ontologies such as for audiovisual services is a challenge. Tools to facilitate this are currently missing.

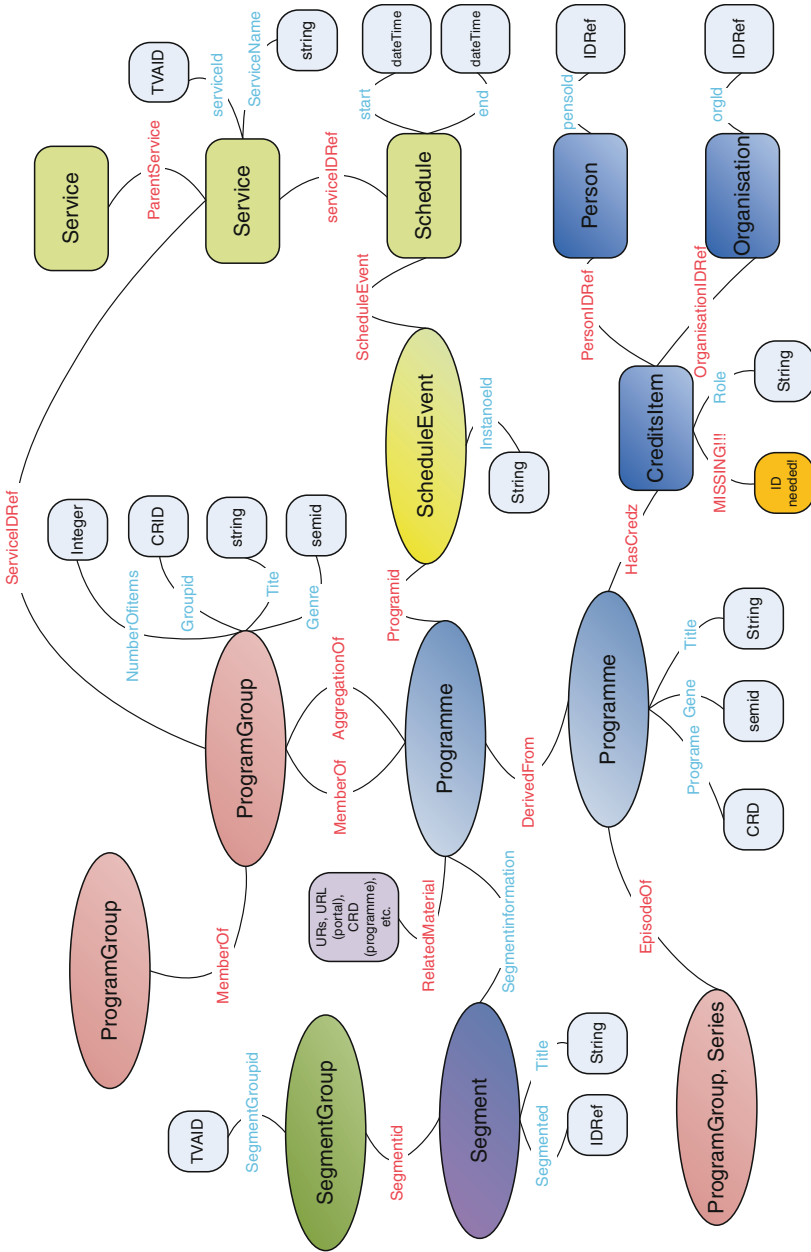


Fig. 21.9 Entity relationship diagram of the TV-Anytime class model

Ontology

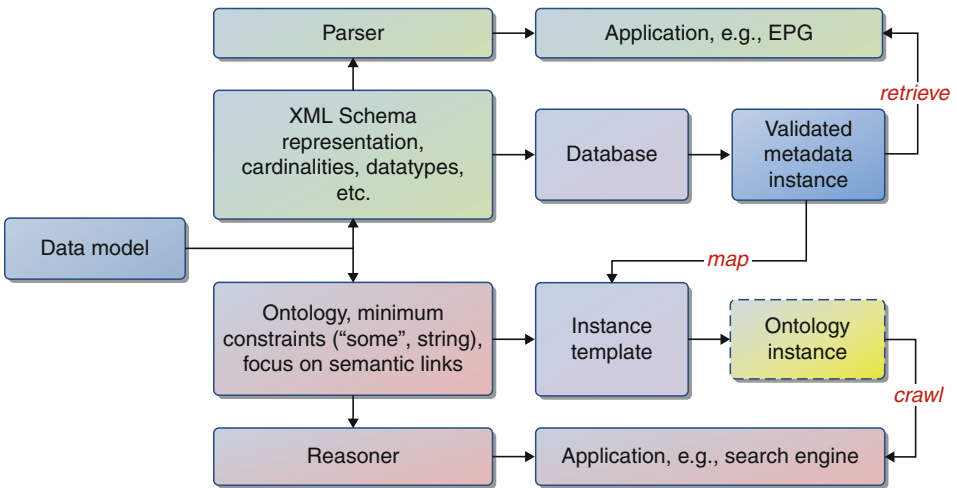
```
tva:hasTitle      a owl:DatatypeProperty ;
                  rdfs:domain tva:Programme ;
                  rdfs:range  xsd:string .
```

Instance

```
_____ tvshows:102587  a tva:Programme ;
                  tva:hasTitle "Tonight's Show" .
```

▣ Fig. 21.10

Example of RDF statement, schema, and instance



▣ Fig. 21.11

Combining XML and RDF/OWL

The main advantages of ontologies for broadcasters are:

- The simplicity of flat statements about resources
- The scalability to create new classes and properties, for example, for customization or particular applications, in a backward compatible manner
- The possibility to infer properties
- The flexibility to use new query approaches.

Some of the disadvantages of ontologies for broadcasters are:

- A steep learning curve
- The danger of confusing concepts and misusing, for example, class and subclasses
- The management of cardinalities
- The nontrivial conversion of XML structures in RDF
- The lack of editing and validating tools

21.2.4.5 A Semantic Representation of Thesauri

Finally, another important part of the TV-Anytime specification is the Classification schemes such as the controlled lists of genres and roles. The EBU has converted some of the TV-Anytime classification schemes into SKOS (Simple Knowledge Organization System), see <http://www.ebu.ch/metadata/ontologies/skos/>.

SKOS is a vocabulary that is very convenient for representing classification schemes with object properties like *broader term* or *narrower term*, *exactMatch*, *narrowMatch*, or *broadMatch*.

As shown in [▶ Fig. 21.12](#), each term of a classification scheme (or thesaurus) is independently subject to a series of statements and is no longer part of a hierarchical XML structure such as used by MPEG-7, TV-Anytime or DVB. Nevertheless, the hierarchical structure can be reconstituted by reasoners as shown in [▶ Fig. 21.13](#), and also include machine-interpretable statements about mapping to other external classification schemes. Ontologies and class models like SKOS are the answer to resolving access to classification scheme terms:

- In MPEG-7, TV-Anytime or DVB, Classification Schemes are defined as hierarchical lists of terms identified by a termID (the access key). Each term has at least a name and a definition. In the XML space, resolving a URI with termID into a term name requires to put in place additional resolving mechanisms (e.g., developing a particular software interface or API).
- In RDF, an object property will point to the SKOS class called “concept” identified by its URI (e.g., the classification scheme locator and termID). If the classification scheme has been imported or can be connected to, all data properties of the concept are directly accessible. This is the lowest but very demonstrative level of “linked data.” Any ontology can therefore refer to any term of a SKOS classification scheme.

Other mechanisms than SKOS could be defined in order to describe classification schemes. However, the need for interoperability requires agreeing on a set of well-defined classes and properties, which SKOS successfully proposes for controlled vocabularies.

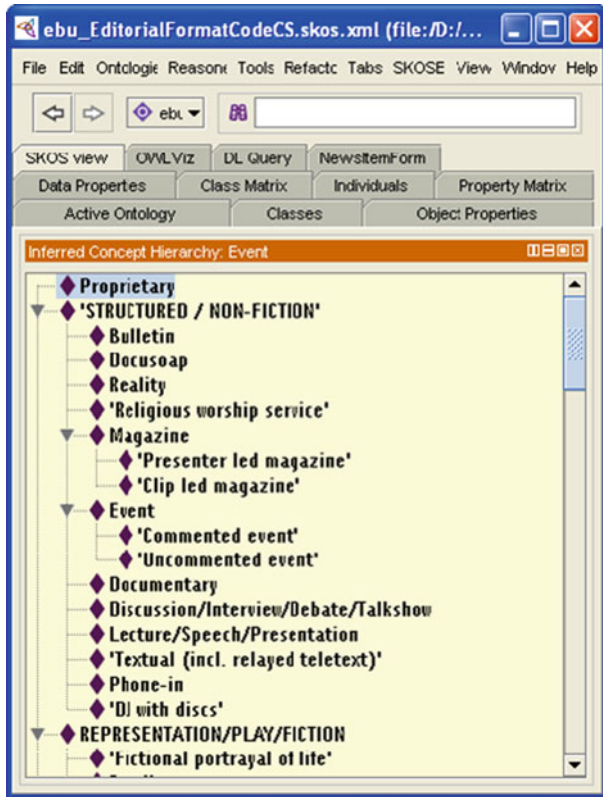
```

ebu:ebu_ContentGenreCS.skos.xml#3.6.8.16.4
    a                skos:Concept ;
    skos:note        "Valid" ;
    skos:historyNote "2007-04-12" ;
    skos:changeNote  "First version" ;
    skos:prefLabel   "Dubstep" ;
    skos:narrowMatch
http://www.ebu.ch/cs/tva/ContentCS.xml#3.6.8.16.4 ;
    skos:broader
ebu:ebu_ContentGenreCS.skos.xml#3.6.8.16 .

```

▶ Fig. 21.12

Extract from EditorialFormatCodeCS



■ Fig. 21.13

Screenshot of a SKOS view of EditorialFormatCodeCS using Protégé

21.2.4.6 The Holy Grail: Agreeing on a Class Model

Standardization groups like the European Broadcasting Union (EBU), the International Press and Telecommunications Committee (IPTC), and W3C Media Annotation Working Group (MAWG) are now paying more attention to the Semantic Web and linked data, generally starting by the “SKOSification” of their classification schemes. More interestingly, there is also an attempt coordinated by EBU to define a common basic class model for audiovisual content. These are the main classes used by some audiovisual schemas (among several others):

- BBC “Programme Model” (<http://www.bbc.co.uk/ontologies/programmes>): brand, series, episode, program, program item, version (of the program), segment, broadcast (event), service, channel, broadcaster (role), person
- CableLabs: asset, chapter, distributor, provider, person, actor, director, producer, studio
- EBUCore (http://tech.ebu.ch/docs/tech/tech3293v1_1.pdf), EBU P-META (http://tech.ebu.ch/docs/tech/tech3295v2_1.pdf) & W3C MAWG: resource, creator, contributor,

- publisher, location, collection/group, fragment/part/segment, concept (classification), person, organization
- ETSI TV-Anytime: program group (including brand, series, etc.), program, segment group, segment, service, schedule, location (broadcast event, schedule event, on-demand program, push download program), person, organization, concept (classification)
 - FRBR, work, expression, manifestation, item, person, corporate body, event, place, concept, object
 - IPTC newsML-G2 (<http://www.iptc.org/cms/site/index.html?channel=CH0111>): news item, part, person, organization, creator, contributor, person, organization, concept
 - ISO/IEC MPEG-7: audiovisual segment, video segment, audio segment, text segment, segment group, audiovisual region, fragment, collection, agent, person, organization, place, event, object
 - PBCore: resource, creator, contributor, publisher, concept (classification)

As can be seen from the examples above, nothing should prevent minimum harmonization but a lack of willingness. To be finalized, this model will also require detailed semantics for every class. Furthermore, several classes are eligible to become subclasses. Of course, the model can be complemented with user-defined classes or a user can utilize only a subset of the above defined classes.

21.2.4.7 Agreeing on Properties

A first level of interoperability is achieved by defining a common set of classes. The effort needs to be repeated on properties. There are two main types of properties in semantic modeling:

- *Object properties* defining relation between classes/objects. *EpisodeOf*, *AggregationOf*, and *MemberOf* are very explicit examples as shown in [▶ Fig. 21.9](#).
- *Data properties* qualifying a class/object, of which typical examples are “Title,” “Identifier,” “Description.”

Properties must be selected properly. The most important criterion consists of defining properties on which queries will be done: What is it that users will or should be looking for? The second criterion is the definition of inverse transitive properties, which, by inference, will enrich the number of triples in stores on which queries will be done, therefore maximizing the chances of positive query hits. In a linked data environment, the third criterion is to reuse existing ontologies defining classes and properties such as FOAF (Friend of a Friend) for persons and contacts. Of course, the choice of such links to existing ontologies shall not prevail upon the efficiency of a solution developed for a particular application within a specific ecosystem. All that matters is the interoperability requirement, which may vary. Linked data also raise issues like persistence and

(e.g., editorial) quality. While in XML, agreeing on properties is more problematic because the model is often closely linked to a particular application, in RDF, properties and classes can complement those in an existing ontology. [Figure 21.14](#) shows a possible high-level class model encompassing the commonalities of the schemas listed above. This may be a first step toward a harmonization of audiovisual content metadata.

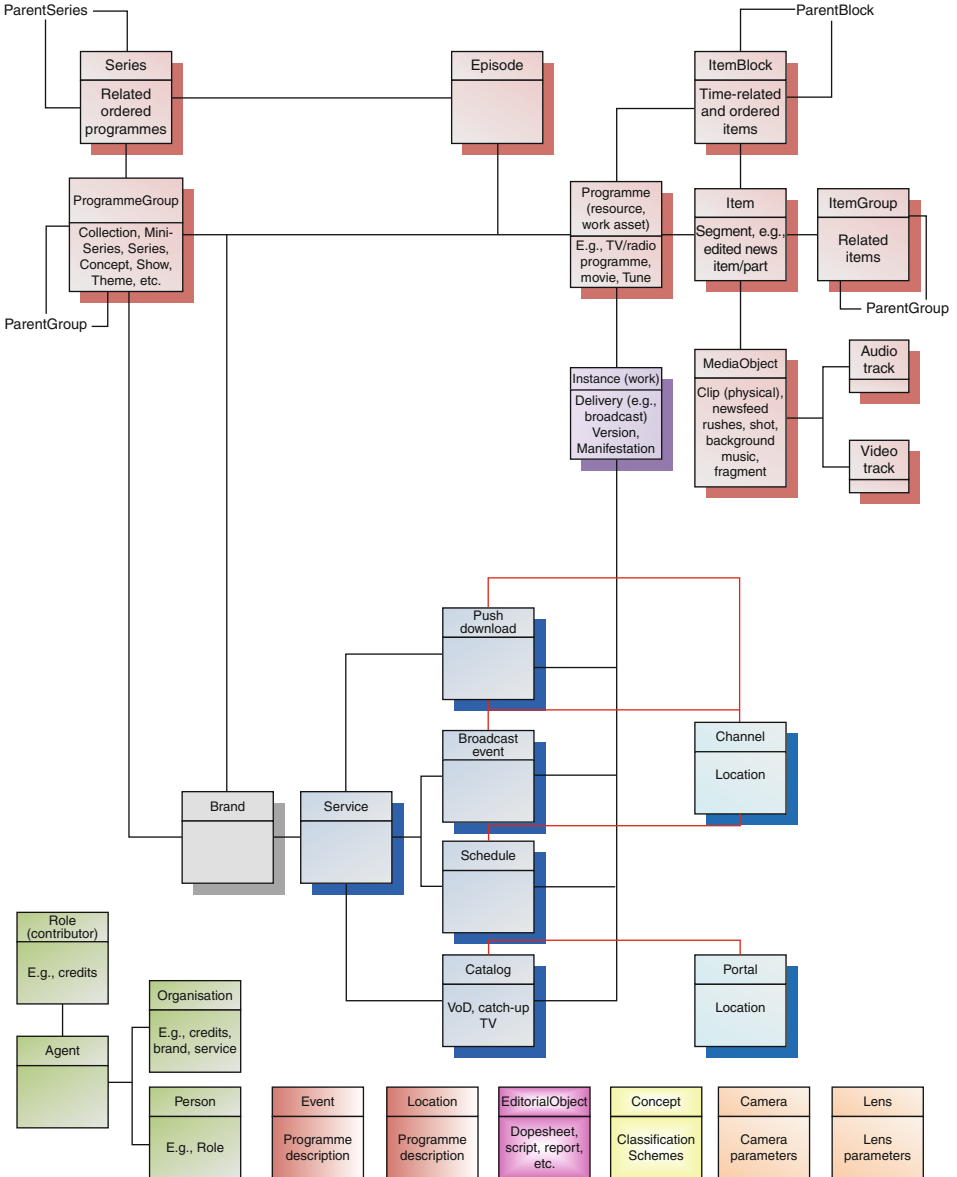


Fig. 21.14
A unified class model?

21.3 Example Applications

In this section, two major domains for the technologies and tools of semantic multimedia are introduced with examples of semantic multimedia technology application: television, and cultural heritage. Section 21.3.1 is partially based on [69] with acknowledgment to the coauthors.

21.3.1 Semantic Television

Television has on the one hand traditionally meant the broadcast industry and on the other hand now incorporates the growing Web-based video domain which is converging with classical broadcast TV in the end device. In this domain, the main atomic object for semantic description is the individual TV program (or video item), while there may be a further higher-level description of the structure of those programs (EPG metadata, or a video playlist). The main challenges in the television (and, by extension, Web video) domain are the scale of the content available and the need for filtering and personalization of the content. The NoTube project (notube.tv) considers three particularly representative scenarios for future television enabled by semantic technology:

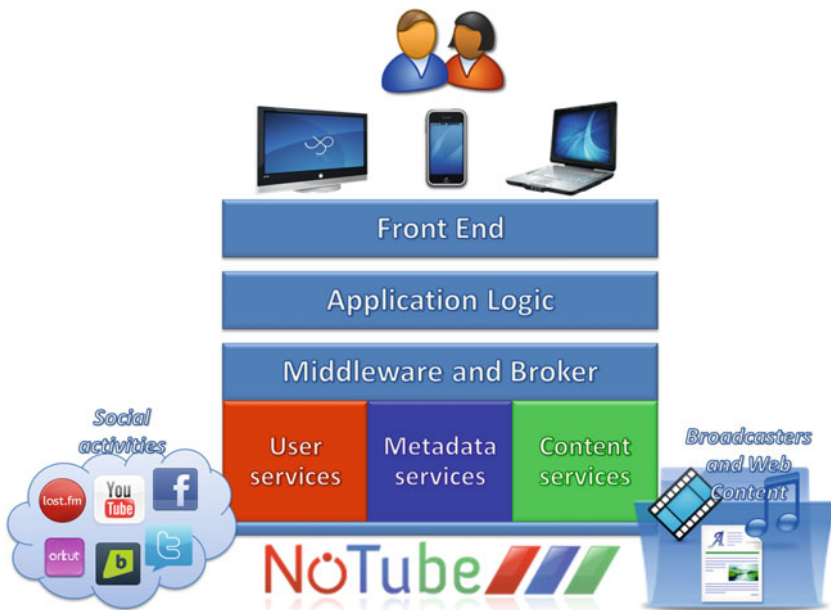
- (a) The RAI demonstrator shows how news programs can be enriched with concepts (people, places, themes) that allow personalized delivery of a news stream and easy browsing to additional information. This demonstrator focuses on the value of **passive personalization of on-demand TV content**.
- (b) The Stoneroos demonstrator enables a user to create an interests profile in a simple fashion, which can be used to generate TV program recommendations within their personal EPG. This demonstrator focuses on the value of a **multi-platform and multilingual platform for personal content and ads**.
- (c) The BBC demonstrator shows how TV can be personalized using Social Web data, facilitating a personalized TV experience without an intrusive user profiling process. This demonstrator focuses on the value of **active personalization of TV which is integrated with the user's Social Web activities**.

To illustrate what is envisaged more generally by semantic TV, Jana is introduced as an example future user of the NoTube infrastructure. She is socially active on the Web and does not see the need to explicitly define her preferences or wait until she has used the recommender system long enough for it to learn her preferences. In the first use case, Jana's recommendations are generated based on her online social activity. In the second use case, Jana is interested in a program and uses the "I would like to know more"-button. Jana then gets information about this program, which contains links to Wikipedia, IMDB, or online information sources. Next to this, she also gets recommendations of related programs. With the "why"-button option Jana can see why each program has been recommended to her. As enriched TV program descriptions are considered, the reasons for recommendations are often based on interesting semantic relations between entities.

For example, when Jana is watching an episode of “True Blood,” this makes her curious about the series, so she picks up her smartphone to find out more about it using the NoTube application. When she presses the “I want to know more”-button the Wikipedia page is shown as well as some recommendations. One of the recommendations is the pilot of the series “Six Feet Under,” which she already knows. She is curious about the reason of the recommendation, so she presses the “why?”-button next to it and sees that both series were created by “Alan Ball” and they share two genres: “black comedy” and “drama.” She is happy to learn that the two series were created by the same man and continues by looking up information about Alan Ball.

The open NoTube TV and Web infrastructure is illustrated in [Fig. 21.15](#). The front end which is what the user sees is any device connected to the Internet and able to consume NoTube services, whether a TV, PC, or mobile device, including a so-called second screen (where a smaller mobile device is used to present auxiliary content in synchronization with the TV signal on the larger screen device). The Application Logic implements the workflow of data and processes which realize the NoTube service to the end device. It relies on the Middleware and Broker layer for this, which makes use of Semantic Web Service technology to dynamically discover and resolve adequate, available services into more complex service workflows.

To enable this, sets of services for users, metadata, and TV content are developed, which are described semantically and mediated by the broker, and in front specific applications are developed to make use of those services to provide the desired



■ Fig. 21.15

NoTube open TV and Web infrastructure

functionalities, for example, user activity capture, and content recommendation. The section considers, from the **user services**, the Beancounter service which harvests user data from online social profiles, which are used to generate a semantic user-interests profile. On the basis of that interests profile, TV program descriptions are analyzed and recommendations are made to the viewer. From the **content services**, the Data Warehouse service collects and enriches EPG data. Existing EPG harvesting services, such as XMLTV, are used to obtain EPG data. The descriptions of TV programs are then enriched by **metadata services**, such as the Named Entity Recognition service, which identifies entities from the linked data cloud. The NoTube vocabulary alignment service identifies links between concepts of different vocabularies.

21.3.1.1 User Activity Capture

The huge amount of interactions that a user performs on the Web represents a powerful and extraordinary source for mining his or her interests and preferences. Even if several attempts already tried to infer a profile of a user starting from his or her behavior on the Web [121–123], the NoTube approach focuses on the opportunities unleashed by the so-called Social and Real-time Web – where users discover, consume and share contents within their social graph, in a real-time manner often using mobile devices. In such environments each user produces a rich and machine-readable flow of activities from which implicit and explicit information regarding his or her preferences can be extracted.

This scenario considers a generic user who holds at least two accounts on different social Web applications: Last.fm and Glue.com. Last.fm tracks the user in listening, sharing, and live events activities. Glue.com acts like a user log, realized as a browser plug-in. Glue.com makes available through a set of Web APIs an exhaustive set of Web resources the user visited, shared, or liked, enriching them with an internal categorization. The following sections show how data are aggregated from these sources, linked to information in several linked data clouds, and how reasoning over the data can make explicit the user's interests in a user profile. The information aggregation is achieved through identity resolution made against different ontologies.

To uniformly represent user activity data of different sources in a single graph, the ATOM Activity Streams in RDF vocabulary (<http://xmlns.notu.be/aaif/>) is used to represent user activities. To determine the objects of activity, a named entity recognition service is used. An alignment service is used to link the objects of different vocabularies, for example, Last.fm artists are linked to DBpedia entities and the BBC Music catalog (<http://www.bbc.co.uk/music>). Vocabularies are defined for TV-related user activities, that is, verbs such as, for example, “watching, reviewing, rating.”

The data generated by the data collection and enrichment process form a potentially huge amount of activities for each individual user. The challenge is to derive general user interests from this set of user activities. This is done by using the DBpedia SKOS vocabulary, which is the semantic counterpart of the Wikipedia Categories. If a user listens to bands or musicians sharing the same subject, then it could be reasonable to infer

that the subject represents an interest for that user. Moreover, the rich and complex SKOS hierarchy of DBpedia allows one to extract a lot of other interesting information. For example, if a user is particularly interested in movies where a particular actor or actress appeared, more information will be available about this in the system, since it is highly probable that DBpedia contains some SKOS subjects describing this. Similarly, if a user listens to bands originating from a specific geographical region then this could be useful to perform recommendations of other bands and artists.

21.3.1.2 Enriched EPG Data

In general, EPG data are produced by broadcast companies. Some broadcast companies, such as the BBC, have made their EPG data machine-readable and publicly available. Other EPG data are harvested from websites using existing tools, such as XMLTV, and converted to a machine-readable format.

Enrichments of EPG metadata are used to provide the end users with extra information about the content in which they are interested. For example, a scheduled broadcast of a movie could have an enrichment that enumerates the main actors together with the pointers to IMDB. Recommender algorithms that fall in the category of content-based filtering algorithms use content descriptions of the items for determining the relevance to the users. To give a simple example, when a user often watches content annotated with the Western concept, then other content annotated with the same or related concepts may be interesting to him or her.

In the NoTube project broadcast data are enriched using the linked data cloud, by linking existing data sources, for example, DBpedia (subject data), Yago (data about people), and IMDB (data about movies) to broadcast metadata. By enriching the EPG data, links to semantic entities in the linked data cloud are added to the metadata of TV programs. The interconnected entities in the linked data cloud allow for finding interesting relationships between entities, for example, that two movies have been made by people that have a common interest in film noir. The relationship between entities is often typed, for example, by SKOS relationships. Since not all relationships between entities are considered interesting, the types of the relationships must be taken into account during the recommendation process. This can be done by using relationships of specific patterns and/or assigning a certain weight to specific relations.

21.3.1.3 Alignment Between Vocabularies

The data sources described above are often already annotated with a fixed set of concepts. For example, EPG data from the BBC (<http://www.bbc.co.uk/programmes>) are annotated with BBC-defined genre hierarchy and IMDB-categorized TV series and Films into a similar set of genres. Vocabularies can be domain-independent, for example, Princeton WordNet (<http://wordnet.princeton.edu>) provides a set of lexical concepts that match words (e.g., in English) and provides semantic relations between those words. Such

vocabularies can be used to annotate domain-specific data. For example, the description of a movie could be a set of WordNet concepts. For some datasets the Semantic Web community already converted the vocabularies and schemas in RDF, like the BBC Programmes ontology (<http://bbc.co.uk/ontologies/programmes>), the TV-Anytime schemas and vocabularies, W3C WordNet. To cover multiple perspectives (extracted) additional genre vocabularies for sources like YouTube are also created.

21.3.1.4 Personalized TV Program Recommendation

Given the availability of the semantically enriched EPG/TV program data, the semantically enriched user activity and interests profile, and the alignment between different vocabularies, the NoTube component Beancounter is able to process the combination of these data to provide a personalized TV program recommendation. The recommendation strategy in NoTube takes a content-based approach, in which the closeness between concepts in a classification scheme (e.g., the DBpedia categorization model) is taken to provide a weighting of the topics with which a program is annotated with respect to the set of topics in the user's profile. To detail this further:

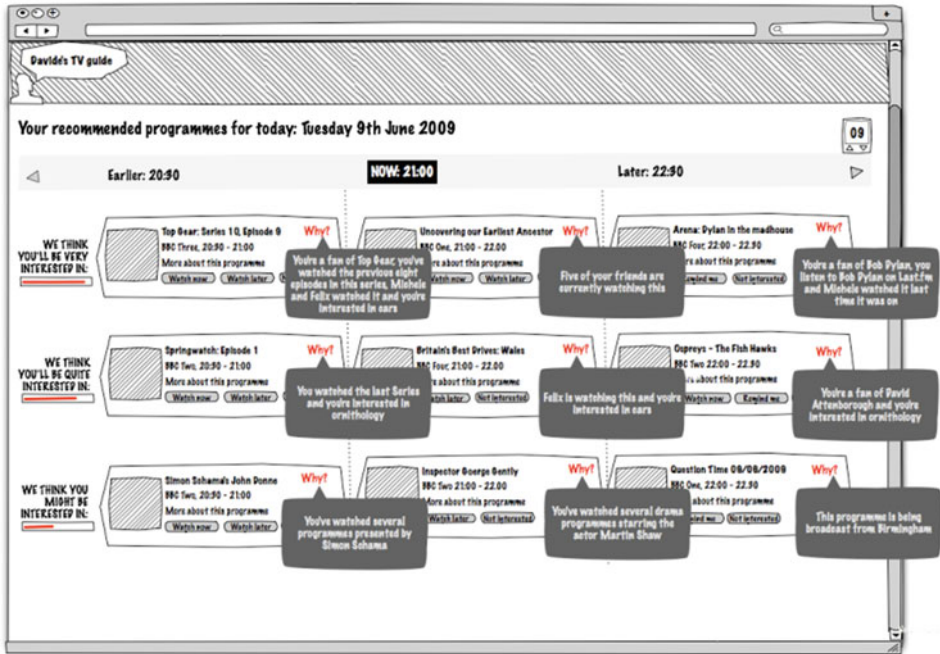
1. Identify weighted sets of DBpedia resources from user activity objects.
2. Compute the distance between DBpedia concepts in the user profile and in the program schedule through a SKOS-based categorization scheme.
3. Choose the matches above a certain threshold for TV program recommendation.

As a result, it should be possible to present the user, through their EPG, for example, a highlighting of TV programs, which should interest them and also to provide some explanation for the recommendation, as shown in the mock-up below of a personalized EPG (► [Fig. 21.16](#)). The same technologies are used in the back-end to enable the other NoTube scenarios, such as a personalized news stream, or pushing personalized advertising.

21.3.2 Semantics in Cultural Heritage

Objects and content in cultural heritage (CH) are both textual and non-textual, interrelated with each other in various ways, and are produced by various organizations and individuals in different ways. As a result, producing, harvesting, aggregating, publishing, and utilizing cultural heritage content on the Web is difficult in many ways. In this section, three major problem areas are covered:

- (1) **Semantics for cultural heritage.** Ontologies and metadata formats are the key components needed in representing CH content on the Semantic Web. Rich ontologies and complex metadata models covering more or less all aspects of human life are needed for representing the semantics of culture for machines.
- (2) **Content creation challenges.** Cultural heritage content is produced in a distributed creation process by various organizations and individuals from different cultures using



■ Fig. 21.16

Mocked-up personalized EPG from NoTube project

different languages. The content is typically very heterogeneous, both in terms of metadata formats and vocabularies/ontologies used. However, from the end user's viewpoint, content should be accessible seamlessly using different languages and vocabularies from different eras, which means that the content should be made semantically interoperable.

- (3) **Semantic eCulture systems.** Semantic computing facilitates searching, linking, and presenting the semantically interlinked, heterogeneous, multi-format, and multilingual CH content. This applies to both professional and layman end users, as well as to machines using CH repositories through service APIs.

Semantic Web technologies provide new solution approaches to all these areas, and cultural heritage (CH) has become an important application domain for semantic technologies. This section presents an overview of issues and solution approaches related to representing ontologies and metadata of cultural heritage, to creating syntactically and semantically interoperable content, and to creating intelligent end-user applications on the Semantic Web.

In journalism and multimedia, content is often collected, described, and searched in terms of the “Five Ws and one H”:

- Who? Who was involved?
- What? What happened and what was involved?

- Where? Where did it take place?
- When? When did it take place?
- Why? Why did it happen?
- How? How did it happen?

In the following, firstly properties of semantic cultural content along these ontological dimensions are discussed.

21.3.2.1 Ontological Dimensions

To answer the who-question, vocabularies, authority files, and ontologies of persons, organizations, and fictive actors have been created. The problems of identifying and describing, for example, persons are well known in, for example, the library domain [69]. For example, similar names are shared by many individuals (e.g., John Smith), names change in time (e.g., when getting married), names are transliterated in different ways in different languages, people use pseudo names and are known by nicknames. An example of an extensive authority system is the Library of Congress Authority Files (<http://authorities.loc.gov>). The Universal List of Authority Names (ULAN) (<http://www.getty.edu/research/conductingresearch/vocabularies/ulan/>) of Getty Foundation is widely used in cultural institutions and Semantic Web systems.

The what-question involves both events that take place and tangible objects that participate in events. Events are a central category in knowledge representation of artificial intelligence (AI) [70], and have been employed in semantic cultural heritage systems, too. Events form the core of the CIDOC CRM system [71], an ontological system for harmonizing cultural heritage and library content. By describing what actually is happening in the real world, heterogeneous content can be harmonized and made interoperable in a deeper semantic sense [73]. In ontologies, such as DOLCE [72] and WordNet [74], events are separated from other ontological concepts. Events remain a difficult concept to represent through an ontology as they are complex and necessarily involve many other concepts in a particular relationship to one another.

For representing tangible objects, cultural heritage thesauri such as the Art and Architecture Thesaurus (AAT) are available. These thesauri make it possible to identify and disambiguate object types from each other and harmonize references to them. Additional ontological descriptions are needed for more refined semantic descriptions of objects. One dimension here is the structure of the object. This involves, for example, describing various part-of relations [74], such as area inclusion, member-of, made-of, and consists-of relations. For example, a material ontology can be used for describing the materials of which objects are made of. A consists-of relation may describe the composition of objects, for example, that legs are part of chairs. Also the function of objects is often important to know, for example, that ships are used for sailing. Such relations are needed, for example, when aggregating related information and objects together in search and recommender systems.

A research area of its own is creating and searching 3D representations of cultural artifacts and buildings [112]. For example, there are 3D models of CH buildings and cities, such as the virtual Kyoto [75], using platforms such as Second Life and Google Earth.

The where-dimension in the CH domain is challenging, because one has to deal with not only modern geographical places, available in resources such as GeoNames and various national geographical gazetteers, but also with historical places that may not even exist today. The Thesaurus of Geographical Names (TGN) is a resource in which lots of historical places can be found. A problem in dealing with historical places is that cultural content is typically indexed (annotated) using historical names (e.g., Carthage) but can be queried using names from different time periods (e.g., modern geography), too. To address the problem of changing boundaries and names of historical places, a model and ontology is presented in [76]. In a more general setting, the concept of places is complex and involves not only geographical information. For example, what does “Germany” actually mean in terms of location, time, and culture?

Time and the when-question are of central importance in the cultural heritage domain that deals with history. A special problem of interest here is that time periods are often imprecise in different ways [77, 78]. Firstly, the time may be exact but not known. For example, an artifact may have been manufactured in a certain day but it is not known exactly when, only an estimate of the year, decade, or century may be known. This kind of uncertainty of time can be modeled, for example, using time intervals and probability theory. Secondly, the time may be fuzzy in nature. For example, a castle may have been built during a longer period (or periods) of time with different intensity, so it is not possible to state exactly when it was actually built. A modeling option here is to use fuzzy sets for representing time. It should be noted also that time periods may not be absolute but are conditioned by places. For example, the notion of the “bronze age” and stylistic periods of art, for example, “art nouveau,” may be different in different countries and cultures.

From a machine viewpoint, formal time representation can be used for reasoning, like in the interval calculus [79], and when matching query time periods with indexing time periods. From the human–computer interaction viewpoint, a key question is how does one perceive uncertain time intervals in information retrieval, that is, when querying with an imprecise time period? For example, querying on “the middle ages,” what time periods should be included in the answer set and how relevant are they?

The question “why” has not been addressed much in CH systems for the Semantic Web. There are, however, several approaches to this. Firstly, it is possible to model causal chains explicitly in annotations. For example, in the history ontology HISTO (<http://www.seco.tkk.fi/ontologies/histo/>), there are some 1,200 historical events of history some of which are related to each other using causal chains. Explicit links or transitive link chains based on them can be then shown to the end user illustrating the why-dimension. A problem here is that there may be disagreements about historical causality and other facts between the historians creating ontologies or annotating the content. Actually, it is not uncommon that knowledge in humanities is based on different opinions. Metadata

can then be used for encoding different opinions, for example, what was the cause of the World War II. Secondly, on a reasoning level, implicit relations between related objects of interest can be explained based on the rules used during reasoning, as customary in some expert systems of artificial intelligence research. For example, in [80], the semantic recommendation links between related artifacts are produced using Prolog rules, and a simple explanation of the reasoning chain in natural language is exposed to the end user.

Semantic CH systems have the potential to address the why-question by exposing and presenting cultural content in novel ways that helps one in understanding cultural phenomena and processes. Semantic techniques and computing can be used as a tool of research for making explicit something useful or new that is only implicitly present in a repository of cultural content. At this point, one enters the field of digital humanities [81]. For example, the idea of associative or relational search, developed for security applications [82], can be used as an approach to answer why-questions. For example, in [83], one can query how two persons are related to each other based on the social network of the ULAN registry of historical persons. Relational search is available also in [84].

Finally, the how-question addresses the problem of describing how things happen(ed). By chaining and relating events with each other and by decomposing them into sub-events, semantics of narrative structures such as stories can be presented. For example, in [85], modeling CH processes and stories using RDF(S) is discussed. In [83], the narrative structures of the epic Kalevala, and the processes of making leather booths, ceramics, and farming have been modeled as narrative structures and interlinked to related CH contents, such as objects in museum collections.

21.3.2.2 Challenges of Content Creation

Cultural heritage content is available in various forms, is semantically heterogeneous, is interlinked, and is published at different locations on the Web. From the end user's viewpoint, it would be useful if content related to some topic, person, location, or other resource could be aggregated and integrated, in order to provide richer and more complete seamless views to contents. This is possible only if the interoperability of heterogeneous content can be obtained on syntactic and semantic levels.

There are two major ways to address interoperability problems: one can either try to prevent them during original content creation or one can try to solve the problems afterward when aggregating content and creating applications. Preventing interoperability problems is the goal of various efforts aiming at developing standards and harmonized ways of expressing content, metadata, and vocabularies as well as best practices for cataloging. The process of producing content can be supported by various shared tools, such as shared ontology services and metadata format repositories.

Although harmonizing content creation would in general be the optimal strategy to address interoperability issues, this is in practice possible only to some extent. As a result, lots of post-processing effort is needed for solving interoperability problems afterward when making existing non-harmonized content syntactically and semantically interoperable.

21.3.2.3 Syntactic and Semantic Interoperability

Syntactic interoperability means that data are represented in similar formats or structures. Syntactic interoperability requires that similar fields are used in metadata structures, and that their values are filled using similar formats. For example, one may demand that the name of a person in two interoperable metadata schemas should be expressed using separate properties “firstName” and “lastName,” and that the name strings used as value are transliterated using the same system. For example, the name of Ivan Aivazovsky, the Russian painter (1817–1900) has 13 different labels in ULAN (Ajvazovskij, Aivazovski, Aiwasoffski, etc.), all correctly transliterated in their own way.

Since Semantic Web content is represented using ontologies and metadata schemas, semantic interoperability issues come in two major forms. First, there is the problem of *schema interoperability*, that is, how two different metadata schemas of similar or different content types can be made mutually interoperable. For example, the “painter” of a painting and the “author” of a novel should somehow be declared semantically related as creators of a piece of art, otherwise all creators cannot be found and related. It is also possible that syntactically similar property names in two schemas have different meaning, which leads to semantic confusion. Second, there is the problem of *vocabulary interoperability*. Here, values of a metadata schema field in content from different organizations may have been taken from different vocabularies that are related, but this relation has not been explicated. For example, a vocabulary may have the concept of “chair” while “sofa” is used in another one. It is also possible that the same label has different interpretations in different vocabularies, for example, “chair” as a piece of furniture, or as a spokesman of an organization. CH concerns various topic areas, such as art, history, handicraft, etc. in which different thesauri and vocabularies are used. Even within a single topic area, different mutually non-interoperable vocabularies may be used.

There seems to be at least three approaches to obtaining metadata schema interoperability. First, a minimal “core” schema can be specified that defines the common parts of all schemas in focus. Then, more refined schemas called applications can be extended from the core by introducing new fields and refining original ones. This approach has been adopted by the Dublin Core (DC) Metadata Initiative. For example, “date” is a DC element that can further be specified as “date published” or “date last modified.” The core elements can be refined or qualified in an interoperable way by formal expressions, which relate the refinement or qualification back to the core element, for example, the relationship between a core property and its refinements can be represented in RDFS using the property `rdfs:subPropertyOf`. An example of a DC application is VRA Core for representing metadata about works of visual culture as well as the images that document them.

Second, it is possible to define a harmonizing ontology or schema that is capable of representing all metadata schemas to be integrated. Semantic interoperability on a schema level is then obtained by transforming the metadata of different forms into this harmonized ontology. A well-known example of this is the CIDOC Conceptual Reference Model (CIDOC CRM) [3], the ISO standard 21127:2006. This model provides definitions and

a formal structure for describing the implicit and explicit concepts and relationships used in cultural heritage documentation. The framework includes 81 classes, such as `crm:Man-MadeObject`, `crm:Place`, and `crm:Time-Span`, and a set of 132 properties relating events and the entities with each other, such as `crm:HasTime-Span` and `crm:IsIdentifiedBy`.

Third, it is possible to transform all metadata into a knowledge representation about the events of the world, as customary in AI. This approach involves developing and using domain ontologies/vocabularies for representing events and objects, not considered in the CIDOC CRM standard focusing on schema semantics [72].

A major area of research in semantic CH applications is the (semi)automatic annotation of contents. If contents are described using texts, then named entity, concept, and relation extraction techniques [87] can be employed first in order to move from literal into concept space. For non-textual multimedia content, for example, images, videos, speech, and music, problems of crossing the semantic gap have to be addressed [94].

21.3.2.4 Semantic eCulture Systems

A major application type of semantic technologies in the CH domain has been semantic portals [87]. Examples of such systems include, for example, MuseumFinland [80] presenting artifacts from various museums, MultimediaN E-Culture demonstrator [84] presenting art and artists from various museums, CultureSampo [83] presenting virtually all kinds of cultural contents (objects, persons, art, maps, narratives, music, etc.), CHIP [88] for personalized mobile access to art collections, and Mobile DBpedia Mobile [89] for mobile access to linked data contents. Systems such as Wikipedia (DBpedia) and Freebase include lots of semantically linked CH content. In addition to systems utilizing Semantic Web technologies, there are even more eCulture sites, portals, and applications on the Web implemented using more traditional technologies. Many of these systems have been reported since 1997 in the Museums and the Web conference series (<http://www.archimuse.com/conferences/mw.html>). A typical eCulture application here is a tailored application for explaining and teaching a particular CH topic with a nice graphical Flash-based user interface.

In this section, research on semantic eCulture portals is described that focus on publishing CH content from the collections of museums, libraries, archives, media organizations, and other sources on the Semantic Web. A common goal in such CH portals is to create a global view over CH collections that are distributed over the Web, as if the collections were a single uniform repository. This idea, developed originally in some national research projects, has also been adapted on an international level in projects such as the European Library and Europeana. These large-scale systems are, however, still based on traditional rather than semantic technologies. There is, however, a demonstration for Europeana's semantic search based on the MultimediaN E-Culture system (<http://eculture.cs.vu.nl/europeana/session/search>).

In order to survive on the Web, a CH portal should be beneficial to both content providers and their customers. We describe below, an ideal “business model” of a semantic CH portal, based on CultureSampo [83], clarifying the challenges and benefits of utilizing semantic technologies in CH portals.

There are two major categories of challenges involved when creating a CH portal: semantic and organizational. First, *semantic challenges* arise from the fact that cultural heritage content is semantically heterogeneous and available in various forms (documents, images, audio tracks, videos, collection items, learning objects, etc.), concern various topics (art, history, handicraft, etc.), is written in different languages, and is targeted at both laymen and experts. Furthermore, the content is semantically interlinked, as depicted in [Fig. 21.17](#).

Second, *organizational challenges* arise from the fact that memory, media, and other organizations and citizens that create the contents work independently according to their own goals and practices, as illustrated in [Fig. 21.18](#). This freedom and independence of publication is essential and empowers the whole Web, but results also in redundant work in content creation, and that interoperability of content between providers cannot be achieved easily. For example, redundant information about cultural persons, places, historical events, etc. has to be collected and maintained in many organizations, because of missing collaboration between organizations. Each organization will have its own database/metadata schema, which cannot be changed.

The Semantic Web–based solution approach to these problems is illustrated in [Fig. 21.19](#), using elements of [Fig. 21.17](#) and [Fig. 21.18](#). The apparatus produces

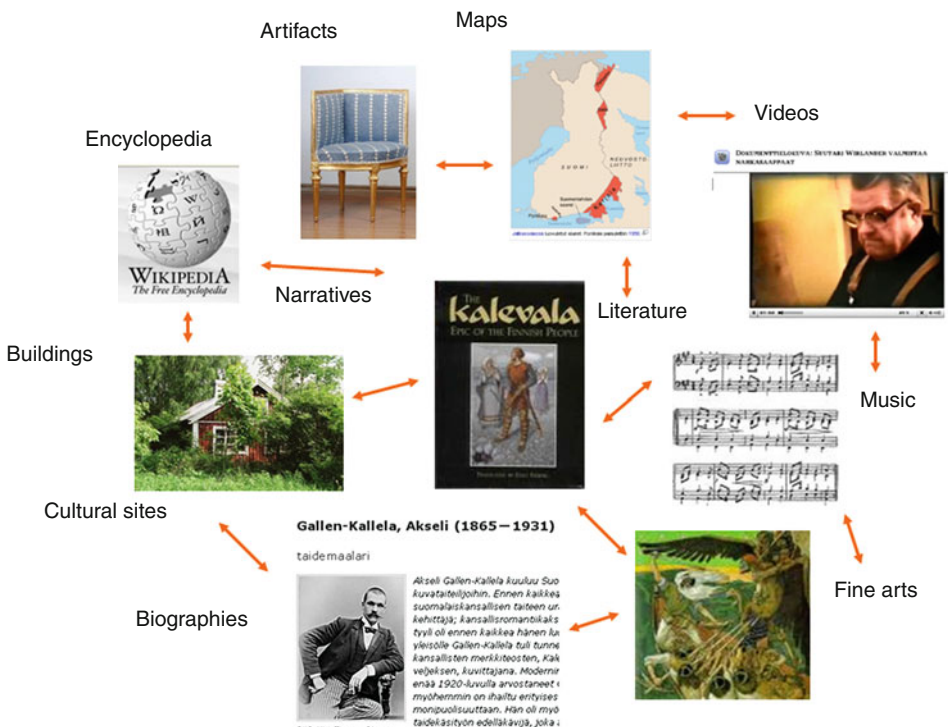


Fig. 21.17

Semantic challenges of CH portals: cultural heritage content comes in many forms and is interlinked



■ Fig. 21.18

Organizational challenge of CH portals: content is produced by independent organizations and individuals for their own purposes with little collaboration

harmonized RDF content for a global knowledge base. In the center are the ontologies forming a conceptual backbone of the system. The collection items around the ontologies are attached to the ontologies by metadata that is produced in terms of harmonized and interlinked metadata schemas and vocabularies. The content providers depicted around the circle, that is, the portal system, publish metadata locally and independently by using shared metadata schemas and ontologies. The result is a large global semantic RDF network linking different contents together in ontologically meaningful ways. When an organization or an individual person submits a piece of (meta)data into the system, the new data get automatically semantically linked to related materials, that is, semantically enriched. At the same time, all related materials get enriched by references to the new piece of knowledge, and through it to other contents. The collaborative business model works, because each additional piece of knowledge is (in the ideal world) beneficial to everybody participating in the system. An additional benefit is that content providers can share efforts in developing the ontology infrastructure, reducing redundant work.

► *Figure 21.19* shows that a semantic CH portal is far more than the portal pages, as seen by the customer on the Web. Firstly, a collaborative ontology infrastructure is needed. This includes a set of cross-domain ontologies, such as artifacts, places, actors, events, time, etc., ontology alignments [90], and a selection of metadata schemas and their alignments.



■ Fig. 21.19

Solution approach – harmonized, distributed production of content, linked together into a global RDF knowledge base

Secondly, a content production and harvesting system is needed for the content providers and the portal for producing and maintaining the content. A most important question here is at what point semantic content is produced: during cataloging by the content providers or afterward when harvesting the content at the portal. The choice depends on the case at hand, but in general high-quality semantic content can be produced best at the organizations producing the content, and shared tools supporting this using the underlying ontology infrastructure can be very useful. For example, in CultureSampo the national FinnONTO infrastructure [91] with its ontology services is used as a basis.

Semantics can be used to provide the end user, both human users and machines, with intelligent services for finding, relating, and learning the right information based on his or her own preferences and the context of using the system. Major functionalities of human user interfaces include:

- Semantic search, that is, finding objects of interest
- Semantic browsing, that is, linking and aggregating content based on their meaning
- Visualization, that is, presenting the search results, contents, and browsing options in useful ways

In the following, these possibilities of providing the end users with intelligent services are briefly explored.

21.3.2.5 Semantic Search

On the Semantic Web, search can be based on finding the concepts related to the documents at the metadata and ontology levels, in addition to the actual text or other features of the data. With such concept-based methods, document meanings and queries can be specified more accurately, which usually leads to better recall and precision, especially if both the query and the underlying content descriptions are concept-based. In practice, semantic search is usually based on query expansion, where a query concept is expanded into its subconcepts or related concepts in order to improve recall. For example, the query “chair” could find “sofas” too, even if the word “chair” is not mentioned in the metadata of sofas. However, care must be taken when expanding queries so that precision of search is not lost. For example, the underlying ontological hierarchies, such as a SKOS vocabulary, may not be transitive leading to problems. For example, if the broader concept of “makeup mirrors” is “mirrors,” and the broader concept of “mirrors” is “furniture,” then searching for furniture would return makeup mirrors, if query expansion is applied. Part-of relations are especially tricky in terms of query expansion. When searching chairs in Europe, also chairs from different countries in Europe can be found. However, “doors” should not be returned when searching for “buildings” even if doors are part of buildings.

A problem of semantic search is mapping the literal search words, used by humans, to underlying ontological concepts, used by the computer. Depending on the application, only queries expressed by terms that are relevant to the domain and content available are successful, other queries result in frustrating “no hits” answers. A way to solve the problem is to provide the end user with explicit vocabularies as facets in the user interface, for example, a subject heading category tree as in Yahoo! and dmoz.org. By selecting a category, related documents are retrieved. If content in semantic search is indexed using language-neutral concept URIs, and their labels are available in different languages, multilinguality can be supported.

A widely employed semantic search and browsing technique in semantic CH portals is view-based or faceted search [95–99]. Here, the user can make several simultaneous selections from orthogonal facets (e.g., object type, place, time, creator). They are exposed to the end user in order to (1) provide him or her with the right query vocabulary, and (2) for presenting the repository contents and search results and the number of hits in facet categories. The number of hits resulting from a category selection is always shown to the user before the selection. This eliminates queries leading to “no hits” dead ends, and guides the user in making the next search selection on the facets. The result set can be presented to the end user according to the facet hierarchies for better readability. This is different from traditional full text search where results are typically presented as a hit list ordered by decreasing relevance. Faceted search is not a panacea for all information

retrieval tasks. A Google-like keyword search interface is usually preferred if the user is capable of expressing his or her information need accurately [101].

Faceted search has been integrated with the idea of ontologies and the Semantic Web [99]. The facets can be constructed algorithmically from a set of underlying ontologies that are used as the basis for annotating search items. Furthermore, the mapping of search items onto search facets can be defined using logic rules. This facilitates more intelligent semantic search of indirectly related items. Methods for ranking the search results in faceted search based on fuzzy logic and probability theory are discussed in [100].

Another search technique now abundant in semantic CH applications is autocompletion. The idea here is to search feasible query word options dynamically as the user types in a query, and to provide the options for him or her to choose from. Semantic autocompletion [102, 103] generalizes this idea by trying to guess, based on ontologies and reasoning, the search concept that the user is trying to formulate after each input character in an input field, or even do the search down to the actual search objects dynamically.

With non-textual cultural documents, such as paintings, photographs, and videos, metadata-based search techniques are a must in practice. However, also content-based information retrieval methods (CBIR) [92], focusing on retrieving images, and multimedia information retrieval (MIR) [93], focusing on retrieving multimedia content, can be used as complementary techniques. Here, the idea is to utilize actual document features (at the data level), such as color, texture, and shape in images, as a basis for information retrieval. For example, an image of Abraham Lincoln could be used as a query for finding other pictures of him, or a piece of music could be searched for by humming it. Tools for navigating, searching, and retrieving 2D images, 3D models, and textual metadata have been developed, for example, in the Sculpteur project (<http://www.sculpteurWeb.org>).

Bridging the “semantic gap” between low-level image and multimedia features and semantic annotations is an important but challenging research theme [94].

21.3.2.6 Semantic Browsing and Recommending

The idea of semantic browsing is to provide the end user with meaningful links to related contents, based on the underlying metadata and ontologies of contents. RDF browsers and tabulators are a simple form of a semantic browser. Their underlying idea has been explicated as the linked data principle proposing that when an RDF resource (URI) is rendered in a browser, the attached RDF links to related resources should be shown. When one of these links is selected, the corresponding new resource is rendered, and so on.

A more developed and general idea is recommender systems [104, 107, 108]. Here, the logic of selecting and recommending related resources can be based on also other principles than the underlying RDF graph. For example, collaborative filtering is based on the browsing statistics of other users. Also logic rules on top of an RDF knowledge base can be used for creating semantic recommendation links and, at the same time, explanations telling the end user why the recommendation link was selected in this context. Recommendations can be based on a user profile of interest and the user’s feedback or browsing log [109, 110].

Semantic recommending is related to relational search, where the idea is to try to search and discover serendipitous semantic associations between different content items. The idea is to make it possible for the end user to formulate queries such as “How is X related to Y” by selecting the end-point resources, and the search result is a set of semantic connection paths between X and Y [83, 84].

The behavior of semantic CH applications should in many cases be dynamic, based on the context of usage [104]. Users are usually not interested in everything found in the underlying content repositories, and would like to get information at different levels of detail (e.g., information for children, professionals or busy travelers). An important aspect of a CH application is then adaptation of the portal to different personal information needs, interests, and usage scenarios, that is, the context of using an application. The context concerns several aspects:

- Personal interests and the behavior of the end user. Material likely to be of interest to the user should be preferred. Techniques such as collaborative filtering could be useful in utilizing other users’ behavior.
- The social environment of the user (e.g., friends and other system users).
- The place and other environmental conditions (e.g., weather) in which the application is used.
- The time of using the system (summer, night, etc.). For example, recommending to the end user that a visit to a beach for a swim during winter may not be wise due to snow, and it would be frustrating to direct him or her to a museum on a Monday when it happens to be closed.
- The computational environment at hand (WiFi, RFID, GPS, ad hoc networks, etc.).

21.3.2.7 Visualization

Visualization is an important aspect of the Semantic Web dealing with semantically complex and interlinked contents [105]. In the cultural heritage domain, maps, timelines, and methods for visualizing complicated and large semantic networks, result sets, and recommendations are of special interest.

Maps are useful in both searching content and in visualizing the results. A widely used approach to using maps in portals is to use mash-up map services based on Google Maps or similar services. For example, lots of Wikipedia articles have location information and can be projected on maps [89]. Maps can also be used as navigational aids.

In the cultural heritage domain, historical maps are of interest of their own. For example, they depict old place names and borders not available anymore in contemporary maps. An approach to visualize historical geographical changes is developed in [106]. Here old maps are laid semitransparently on top of the contemporary maps and satellite images of Google Maps, as a kind of historical lens. At the same time, articles from Wikipedia and photos from services like Panoramio, as well as objects from museum collections can be visualized on top of maps, giving even more historical and contemporary perspective to the contents.

Maps are very usable in mobile phones and navigation systems. Many modern phones include not only GPS for positioning, but also a compass for orientation. In some augmented reality systems, it is possible to point the camera of the device in a direction and get information about the nearby objects there. An example of this type of system is Wikitude (<http://www.wikitude.org>).

Another important dimension for visualizing cultural content is time. A standard approach for temporal visualization is to project objects of interest on a timeline. A generic mash-up tool for creating timelines is the Simile timeline (<http://simile.mit.edu/timeline/>). A timeline can be used both for querying and for visualizing and organizing search results.

21.3.2.8 Cultural Heritage as Web Services

The Semantic Web facilitates reusing and aggregating contents through Web APIs [83]. A starting point for this is to publish the CH repository as a SPARQL end point. It is also possible to develop higher-level services for querying the RDF store. Both traditional Web Services and lightweight mash-ups based on AJAX and REST can be used here. Using the mash-up approach, the functionalities can be used in external applications with just a few lines of JavaScript code added on the HTML level.

The possibility of reusing semantic CH content as a service is an important motivator for organizations to join CH portal projects. In this way, one can not only get more visibility to one's content through a larger portal, but also enrich one's own content with others' related content, and can use the enriched content back for other applications.

21.4 Related Resources Including Key Papers

21.4.1 Multimedia Ontologies, Annotation, and Analysis

Semantic Multimedia. Staab, S., Scherp, A., Arndt, R., Troncy, R., Grzegorzek, M., Saathoff, C., Schenk, S., Hardman, L.: Semantic multimedia. In: Reasoning Web: Fourth International Summer School, Venice, Italy, 7–11 September 2008. Tutorial Lectures. Springer, pp. 125–170 (2008).

In this paper, issues of semantics in multimedia management are dealt with, covering the representation of multimedia metadata using Semantic Web ontologies; the interpretation of multimedia objects by various means of reasoning; the retrieval of multimedia objects by means of low- and high-level (semantic) representations of multimedia; and the further processing of multimedia facts in order to determine provenance, certainty, and other meta-knowledge aspects of multimedia data.

Enquiring MPEG-7-based multimedia ontologies. Dasiopoulou, S., Tzouvaras, V., Kompatsiaris, I., Strintzis, M.G.: Enquiring MPEG-7-based multimedia ontologies. *Multimed Tools Appl* 46(2–3) (January 2010).

Machine-understandable metadata form the main prerequisite for the intelligent services envisaged in a Web, which going beyond mere data exchange and provides for effective content access, sharing, and reuse. MPEG-7, despite providing a comprehensive set of tools for the standardized description of audiovisual content, is largely compromised by the use of XML that leaves the largest part of the intended semantics implicit. Aspiring to formalize MPEG-7 descriptions and enhance multimedia metadata interoperability, a number of multimedia ontologies have been proposed. Though sharing a common vision, the developed ontologies are characterized by substantial conceptual differences, reflected both in the modeling of MPEG-7 description tools as well as in the linking with domain ontologies. Delving into the principles underlying their engineering, a systematic survey of the state of the art MPEG-7-based multimedia ontologies is presented, and issues highlighted that hinder interoperability as well as possible directions toward their harmonization.

COMM: A Core Ontology for Multimedia Annotation. Arndt, R., Troncy, R., Staab, S., Hardman, L.: COMM: a core ontology for multimedia annotation. In: Staab, S., Studer, R. (eds.), *Handbook on Ontologies*, 2nd edn. International Handbooks on Information Systems. Springer Verlag, pp. 403–421 (2009).

This chapter analyzes the requirements underlying the semantic representation of media objects, explains why the requirements are not fulfilled by most semantic multimedia ontologies and presents COMM, a core ontology for multimedia, that has been built reengineering the current de facto standard for multimedia annotation, that is, MPEG-7, and using DOLCE as its underlying foundational ontology to support conceptual clarity and soundness as well as extensibility toward new annotation requirements.

A Description Logic for Image Retrieval. Di Sciascio, E, Donini, F.M., Mongiello, M.: A description logic for image retrieval. In: *Proceedings of AI*IA*, September 1999.

This paper presents a Description Logic–based language that enables the description of complex objects as compositions of simpler artifacts for the purpose of semantic image indexing and retrieval. An extensional semantics is provided, which allows for the formal definition of corresponding reasoning services.

Ontological inference for image and video analysis. Town, C.: Ontological inference for image and video analysis. *Mach Vis Appl* 17(2), 94–115 (2006).

Though focusing solely on probabilistic aspects of the imprecision involved in image and video analysis, the paper elaborates insightfully on the individual limitations of ontological and Bayesian inference, and proposes an iterative, goal-driven hypothesize-and-test approach to content interpretation.

21.4.2 Broadcaster Artifacts Online

21.4.2.1 Vocabularies and Ontologies

- **BBC programmes ontology.** This ontology aims at providing a simple vocabulary for describing programs. It covers brands, series (seasons), episodes, broadcast events,

broadcast services, etc. The data at <http://www.bbc.co.uk/programmes> are annotated using this ontology. <http://bbc.co.uk/ontologies/programmes/>

21.4.2.2 Metadata Schemas

- **TV-Anytime.** TV-Anytime is a set of specifications for the controlled delivery of multimedia content to a user's personal device (Personal Video Recorder (PVR)). It seeks to exploit the evolution in convenient, high-capacity storage of digital information to provide consumers with a highly personalized TV experience. Users will have access to content from a wide variety of sources, tailored to their needs and personal preferences. <http://www.etsi.org/Website/technologies/tvanytime.aspx>

21.4.2.3 Semantic Television

NoTube: making the Web part of personalized TV. Schopman, B., Brickley, D., Aroyo, L., van Aart C., Buser, V., Siebes, R., Nixon, L., Miller, L., Malaise, V., Minno, M., Mostarda, M., Palmisano, D., Raimond, Y.: NoTube: making the Web part of personalized TV. In: Proceedings of the WebSci10: Extending the Frontiers of Society Online, April 2010.

The NoTube project aims to close the gap between the Web and TV by semantics. Bits and pieces of personal and TV-related data are scattered around the Web. NoTube aims to put the user back in the driver's seat by using data that are controlled by the user, for example, from Facebook and Twitter, to recommend programs that match the user's interests. By using the linked data cloud, semantics can be exploited to find complex relations between the user's interests and background information on programs, resulting in potentially interesting recommendations.

21.4.3 Cultural Heritage Artifacts Online

21.4.3.1 Vocabularies and Ontologies

- **Art and Architecture Thesaurus (AAT).** A hierarchical vocabulary of around 34,000 records, including 134,000 terms, descriptions, bibliographic citations, and other information relating to fine art, architecture, decorative arts, archival materials, archaeology, and other material culture. http://www.getty.edu/research/conducting_research/vocabularies/aat/
- **Thesaurus of Geographical Names (TGN).** A hierarchical vocabulary of around 895,000 records, including 1.1 million names, place types, coordinates, and descriptive notes, focusing on places important for the study of art and architecture. http://www.getty.edu/research/conducting_research/vocabularies/tgn/

- **Universal List of Artist Names (ULAN).** A vocabulary of around 162,000 records, including 453,000 names and biographical and bibliographic information for artists, architects, firms, shops, and art repositories, including a wealth of variant names, pseudonyms, and language variants. http://www.getty.edu/research/conducting_research/vocabularies/ulan/
- **Iconclass.** A classification system designed for art and iconography. It is the most widely accepted scientific tool for the description and retrieval of subjects represented in images (works of art, book illustrations, reproductions, photographs, etc.) and is used by museums and art institutions. <http://www.iconclass.nl/>
- **Library of Congress Subject Headings (LCSH).** A very large subject classification system for libraries, available also in SKOS. <http://id.loc.gov/authorities/>

21.4.3.2 Metadata Schemas

- **Dublin Core.** The Dublin Core Metadata Initiative, or “DCMI,” is an open organization engaged in the development of interoperable metadata standards that support a broad range of purposes and business models. <http://dublincore.org/>
- **CIDOC CRM.** The CIDOC Conceptual Reference Model (CRM) provides definitions and a formal structure for describing the implicit and explicit concepts and relationships used in cultural heritage documentation. <http://www.cidoc-crm.org/>

21.4.3.3 Semantic eCulture Systems Online

- **MuseumFinland.** A semantic portal aggregating artifact collections from several museums [80]. The content comes from Finnish museums and the system interface is in Finnish (with an English tutorial). <http://www.museosuomi.fi/>
- **MultimediaN eCulture Demonstrator.** This cultural search engine gives access to artworks from several museum collections using several large vocabularies [84]. <http://e-culture.multimedian.nl/demo/session/search>
- **CultureSampo.** A semantic portal aggregating cultural content of different kinds from tens of different organizations, Web sources, and public [83]. The content comes from Finnish and some international sources, and the user interface supports Finnish, English, and Swedish. <http://www.kulttuurisampo.fi/>

21.5 Future Issues

Semantic technologies are seen as being nearly ready for mainstream adoption as the first decade of the twenty-first century draws to an end. While the situation with respect to textual content is quite mature, non-textual media present additional challenges

to the technology adoption. As a result, the wider adoption of semantic multimedia may first follow the breakthrough of other applications of semantic technology – for example, knowledge management, data integration, semantic search – applied to textual content.

The new challenges being faced by the media industry – the scale and complexity of media being produced and shared – act as a market driver for technological advances in the semantic multimedia field. Online media in particular needs improved retrieval, adaptation, and presentation if content owners are to win market share in a broad and overfilled market. A few media organizations have begun to lead the way in using and demonstrating semantics, for example, the BBC has begun to publish its online content with RDF.

The arts – that is, cultural heritage – are another sector in which semantics are gaining traction. Museums, for example, have large amounts of metadata about their collections, which cannot be easily interpreted or reused due to non-digital, non-semantic, and proprietary approaches. Again, some pioneers, such as Rijksmuseum in Amsterdam, are taking the first steps to digitize and annotate their collections and explore the new possibilities which are realized.

The media, arts, and entertainment sector looks on semantics as a clear future solution for their problems with large scales of heterogeneous non-textual content, and for the emerging challenges in realizing attractive and competitive content offers on a ubiquitous Web with millions of content channels. The cost of creating the semantic data tends to be larger at present than the benefits gained from its creation, so while the potential benefit from semantics will continue to grow as Web media becomes more ubiquitous (making a Unique Selling Point ever more critical for a content owner and provider), the actual costs of semantics must still fall through improved, more automated content annotation tools and approaches. Let us look at trends and technology in two specific target areas for semantic multimedia.

IP Television: IP Television refers to the convergence of Internet and Television, which is also happening outside of the television set (e.g., also Web-based TV, Mobile TV). Currently, it is focused on new types of services around television such as EPGs, programming on demand, and live TV pause. An emerging trend in IPTV is toward Web integration through widgets, which are lightweight self-contained content items that make use of open Web standards (HTML, JavaScript) and the IP back-channel to communicate with the Web (typically in an asynchronous manner). Yahoo! and Intel, for example, presented their Widget Channel at the CES in January 2009, where Web content such as Yahoo! news and weather, or Flickr photos, could be displayed in on-screen widgets on TV. Sony and Samsung will go to market in 2010 with Internet-enabled televisions. A 2009 survey found that there should be a gradual but steady uptake of TV Internet usage with “the mass market inflection point occurring over the next 3–5 years” (from http://oregan.net/press_releases.php?article=2009-01-07). Parallel to this, research into semantic IPTV applications and solutions is being established in academic and industry labs. A key focus for semantics is the formal description of the programming and user interests to provide for a better personalization of the TV experience (EU project NoTube, <http://www.notube.tv>, 2009–2012) as well as formal description of networks and content to enable a better delivery of complex services (myMedia, <http://www.semantic-iptv.de>).

A major barrier to uptake by broadcasters and content providers is the lack of support for semantic technology in the legacy broadcast systems. Shifts in the provider-side IT infrastructure to Internet-based (even cloud-based) infrastructures should give an opening for the introduction of semantics into the production systems of the television and media companies. Vocabularies and technologies will need to converge on specific standards to encourage industry acceptance, as discussed in this chapter's section on broadcasting, which should emerge in this next "uptake" period. As Internet-TV reaches the mass market point (possibly by 2014), companies will seek Unique Selling Points for their products and services, which will drive the incorporation of semantic technologies into IPTV infrastructures and packages.

Virtual Worlds and 3D: The third dimension has always been a part of human perception, but in the digital world it has had a shorter existence. Today, on the other hand, computers are capable of rendering highly complex 3D scenes, which can even be mistaken for real by the human eye. 3DTV is on the cusp of market introduction. A new IT segment must deal with the capturing of 3D objects, their manipulation and management, in application domains from health care to cultural heritage.

Challenges in the 3D technology domain include how to describe 3D objects for their indexing, storage, retrieval, and alteration. Semantics provide a means to improve the description, search, and reuse of complex 3D digital objects. Awareness of the value and potential use of this technology in the 3D media community is at an early stage [111]. It is being promoted to industry through initiatives like FOCUS K3D (<http://www.focusk3d.eu>), which has application working groups for the domains of medicine and bioinformatics, gaming and simulation, product modeling, and archaeology. A survey on the state of the art in cultural heritage [112] notes that progress is being made on standardized metadata schemes and ontologies; current limitations relate to methodologies and the lack of specialized tools for 3D knowledge management, yet this could be addressed in the short to medium term.

Virtual worlds are a natural extension of 3D technology into reflecting the perceptive realities of one's own world and have also found applicative usage in domains such as medicine, social analysis, education, and eCommerce. Making virtual worlds "react" more realistically to actions and activities performed by the actors of that world requires a (semantic) understanding of the objects rendered in the world and the events that (can) occur between them. There is also a trend to more closely couple real and virtual worlds through (real world) sensors, which generate data streams to cause the virtual world to reflect the real in near-real time. This leads to a requirement to handle increasing scales of heterogeneous, dirty data for information extraction and actionable inference within the virtual world.

As in the 3D technology field, barriers to use of semantic technologies lie in the need to agree on the vocabularies and schema for descriptions of the worlds (which now goes beyond the form of objects, and encapsulates what can be done with them, how they react to external events, etc.), as well as the availability of appropriate tools for the creation and maintenance of semantic virtual worlds. Hence, it is likely that semantics will first need to experience wide uptake in 3D technology systems before it also further develops into

a technology for virtual worlds in the medium to long term. Projects such as *Semantic Reality* (<http://www.semanticreality.org>) provide exciting longer-term visions of a virtual world tightly connected to the real world, with trillions of sensors able to ensure a close correlation between both [113].

Such visions of intelligent media and even intelligent worlds will be built on the blocks of semantic multimedia technology discussed in this chapter, once key barriers to uptake are overcome. In particular, semantic annotation of non-textual media remains a significant barrier.

Foundational technologies to (semi)automatically annotate non-textual resources have been investigated by the multimedia semantics community, which spans more broadly the areas of computer vision and multimedia analysis. These areas provide means to analyze visual streams of information with the help of low-level feature extraction, object detection or high-level event inferencing. Despite promising advances, approaches that can be generically and efficiently applied to automate annotation across media still remain to be defined. In contrast to textual resources, which are annotated automatically to a large extent, non-textual media semantic annotation heavily relies on human input, thus being associated to significant costs.

The area of computer vision provides methods to make visual resources eligible for machines. Recent years have seen considerable advancement in the range of things that are detectable in still and moving images. This includes object detection scaling up to a considerable amount of different objects for some tools, to object tracking in, for example, surveillance videos. All of these approaches try to derive meaning from low-level features (like color histogram, motion vectors, etc.) automatically. Despite constant advances, these tools are still not capable of exploiting the full meaning of visual resources as not all meaning is localized in the visual features and needs human interpretation. Two ways are currently followed in current research: The first one is to provide rich human annotations as training data for future automated analysis. The second one relies purely on analysis of raw content, which only performs well for specialized domains and settings in which relevant concepts can be easily recognized. Richer semantics, capturing implicit features and meaning derived from humans cannot be extracted in this manner. Present trends put therefore the human more and more into the loop by lowering the entry barrier for his or her participation. This is done by adopting Web2.0 or game-based approaches to engage users in the annotation of visual resources. Recent approaches, for example, try to support automatic analysis with tagging or vice versa. What is still missing are approaches that are capable of exploiting more high-level features also in visual resources of lower quality and which can be adapted across domains.

Hence, in the foreseeable future, multimedia analysis has still to be supported by end users to a great extent. This is why recent years have seen a huge growth in available annotation tools, which allow manual or semiautomatic annotation of visual resources. These approaches are either targeted at the support of analysis approaches to provide training data or as a means for users to organize media. These approaches show a varying complexity: While some allow users to express complex statements about visual resources, others enable the provision of tags. Some approaches apply annotation or tag propagation and offer support based on

previously supplied annotations. Most of these approaches are still not mature and are only applied in research. While approaches based on (complex) ontologies exist, some of them are not suitable for most end users. At the other side of the spectrum, tagging-based approaches are not suitable to capture all semantics in visual resources. What is still needed are tools that allow one to capture subjective views of visual resources and combine these views to deliver a consolidated objective view that can represent a view which holds across users. While tagging-based approaches are proven to ease large-scale uptake, motivating users to provide more meaningful annotations is still an issue.

21.6 Cross-References

Future Trends

References

1. Deerwater, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. *J. Am. Soc. Inf. Sci.* **41**(6), 391–407 (1990)
2. Bürger, T., Hausenblas, M.: Why real-world multimedia assets fail to enter the semantic web. In: Proceedings of the Semantic Authoring, Annotation and Knowledge Markup Workshop (SAAKM 2007) located at the Fourth International Conference on Knowledge Capture (KCap 2007), Whistler. CEUR Workshop Proceedings 289. CEUR-WS.org (2007)
3. MPEG-7: Multimedia content description Interface. Standard No. ISO/IEC 15938 (2001)
4. van Ossenbruggen, J., Nack, F., Hardman, L.: That obscure object of desire: multimedia metadata on the web (part I). *IEEE Multimed.* **11**(4), 38–48 (2004)
5. Nack, F., van Ossenbruggen, J., Hardman, L.: That obscure object of desire: multimedia metadata on the web (part II). *IEEE Multimed.* **12**(1), 54–63 (2005)
6. Troncy, R., Carrive, J.: A reduced yet extensible audio-visual description language: how to escape from the MPEG-7 bottleneck. In: Proceedings of the Fourth ACM Symposium on Document Engineering (DocEng 2004), Milwaukee (2004)
7. Troncy, R., Bailer, W., Hausenblas, M., Hofmair, P., Schlatte, R.: Enabling multimedia metadata interoperability by defining formal semantics of MPEG-7 profiles. In: Proceedings of the First International Conference on Semantics and Digital Media Technology (SAMT 2006), Athens, pp. 41–55 (2006)
8. Garcia, R., Celma, O.: Semantic integration and retrieval of multimedia metadata. In: Proceedings of the Fifth International Workshop on Knowledge Markup and Semantic Annotation (SemAnnot 2005), Galway, pp. 69–80 (2005)
9. Hunter, J.: Adding multimedia to the semantic web – building an MPEG-7 ontology. In: Proceedings of the First International Semantic Web Working Symposium (SWWS 2001), Stanford, pp. 261–281 (2001)
10. Tsinaraki, C., Polydoros, P., Christodoulakis, S.: Interoperability support for ontology-based video retrieval applications. In: Proceedings of the Third International Conference on Image and Video Retrieval (CIVR 2005), Dublin, pp. 582–591 (2005)
11. Troncy, R., Celma, Ó., Little, S., García, R., Tsinaraki, C.: MPEG-7 based multimedia ontologies: interoperability support or interoperability issue? In: SAMT 2007: Workshop on Multimedia Annotation and Retrieval enabled by Shared Ontologies (MARESO 2007), Genoa (2007)
12. Lagoze, C., Hunter, J.: The ABC ontology and model (v3.0). *J. Digit. Inf.* **2**(2) (2001)

13. Hunter, J.: Enhancing the semantic interoperability of multimedia through a core ontology. *IEEE Trans. Circuits Syst. Video Technol.* **13**(1), 49–58 (2003)
14. Hunter, J., Little, S.: A framework to enable the semantic inferencing and querying of multimedia content. *Int. J. Web Eng. Technol.* **2**(2/3), 264–286 (2005) (Special issue on the Semantic Web)
15. Pease, A., Niles, I., Li, J.: The suggested upper merged ontology: a large ontology for the semantic web and its applications. In: *Working Notes of the AAAI-2002 Workshop on Ontologies and the Semantic Web*, Edmonton (2002)
16. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L.: Sweetening ontologies with DOLCE. In: *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2002)*, Siguenza, pp. 166–181 (2002)
17. Polydoros, P., Tsinaraki, C., Christodoulakis, S.: GraphOnto: OWL-based ontology management and multimedia annotation in the DS-MIRF framework. *J. Digit. Inf. Manag.* **4**(4), 214–219 (2006)
18. Tsinaraki, C., Polydoros, P., Christodoulakis, S.: Interoperability support between MPEG-7/21 and OWL in DS-MIRF. *Trans. Knowl. Data Eng.* **19**(2), 219–232 (2007) (Special issue on the Semantic Web Era)
19. Garcia, R., Gil, R., Delgado, J.: A web ontologies framework for digital rights management. *J. Artif. Intell. Law* **15**, 137–154 (2007)
20. Garcia, R., Gil, R.: Facilitating business interoperability from the semantic web. In: *Proceedings of the Tenth International Conference on Business Information Systems (BIS 2007)*, Poznan, pp. 220–232 (2007)
21. Troncy, R.: Integrating structure and semantics into audio-visual documents. In: *Proceedings of the Second International Semantic Web Conference (ISWC 2003)*, Sanibel Island. *Lecture Notes in Computer Science*, vol. 2870, pp. 566–581. Springer, Berlin (2003)
22. Isaac, A., Troncy, R.: Designing and using an audio-visual description core ontology. In: *Workshop on Core Ontologies in Ontology Engineering at the 14th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2004)*, Whittlebury Hall (2004)
23. Bloehdorn, S., Petridis, K., Saathoff, C., Simou, N., Tzouvaras, V., Avrithis, Y., Handschuh, S., Kompatsiaris, Y., Staab, S., Strintzis, M.G.: Semantic annotation of images and videos for multimedia analysis. In: *Proceedings of the Second European Semantic Web Conference (ESWC 2005)*, Heraklion. *Lecture Notes in Computer Science*, vol. 3532, pp. 592–607. Springer, Berlin (2005)
24. Hollink, L., Worring, M., Schreiber, A.Th.: Building a visual ontology for video retrieval. In: *Proceedings of the 13th ACM International Conference on Multimedia (MM 2005)*, Hilton (2005)
25. Vembu, S., Kiesel, M., Sintek, M., Bauman, S.: Towards bridging the semantic gap in multimedia annotation and retrieval. In: *Proceedings of the First International Workshop on Semantic Web Annotations for Multimedia (SWAMM 2006)*, Edinburgh (2006)
26. Halaschek-Wiener, C., Golbeck, J., Schain, A., Grove, M., Parsia, B., Hendler, J.: Annotation and provenance tracking in semantic web photo libraries. In: *Proceedings of International Provenance and Annotation Workshop (IPAW 2006)*, Chicago, pp. 82–89 (2006)
27. Chakravarthy, A., Ciravegna, E., Lanfranchi, V.: Aktivemedia: cross-media document annotation and enrichment. In: *Poster Presentaiton at the Proceedings of Fifth International Semantic Web Conference (ISWC 2006)*, Athens, GA. *Lecture Notes in Computer Science*, vol. 4273. Springer, Berlin (2006)
28. Petridis, K., Bloehdorn, S., Saathoff, C., Simou, N., Dasiopoulou, S., Tzouvaras, V., Handschuh, S., Avrithis, Y., Kompatsiaris, I., Staab, S.: Knowledge representation and semantic annotation of multimedia content. *IEE Proc. Vis. Image Signal Process.* **153**, 255–262 (2006) (Special issue on Knowledge-Based Digital Media Processing)
29. Simou, N., Tzouvaras, V., Avrithis, Y., Stamou, G., Kollias, S.: A visual descriptor ontology for multimedia reasoning. In: *Proceedings of the Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS 2005)*, Montreux (2005)
30. Saathoff, C., Schenk, S., Scherp, A.: Kat: the k-space annotation tool. In: *Poster Session, International Conference on Semantic and Digital*

- Media Technologies (SAMT 2008), Koblenz (2008)
31. Arndt, R., Troncy, R., Staab, S., Hardman, L., Vacura, M.: COMM: designing a well-founded multimedia ontology for the web. In: Proceedings of Sixth International Semantic Web Conference (ISWC 2007), Busan. Lecture Notes in Computer Science, vol. 4825, pp. 30–43. Springer, Berlin (2007)
 32. Saathoff, C., Scherp, A.: Unlocking the semantics of multimedia presentations in the web with the multimedia metadata ontology. In: Proceedings of 19th International Conference on World Wide Web (WWW 2010), Raleigh, pp. 831–840 (2010)
 33. Smeulders, A.W.M., Worring, M., Santini, S., Gupta, A., Jain, R.: Content-based image retrieval at the end of the early years. *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(12), 1349–1380 (2000)
 34. Rao, A., Jain, R.: Knowledge representation and control in computer vision systems. *IEEE Expert* **3**, 64–79 (1988)
 35. Draper, B., Hanson, A., Riseman, E.: Knowledge-directed vision: control, learning and integration. *IEEE* **84**(11), 1625–1681 (1996)
 36. Snoek, C., Huurnink, B., Hollink, L., Rijke, M., Schreiber, G., Worring, M.: Adding semantics to detectors for video retrieval. *IEEE Trans. Multimed.* **9**(5), 975–986 (2007)
 37. Hauptmann, A., Yan, R., Lin, W.H., Christel, M., Wactlar, H.: Can high-level concepts fill the semantic gap in video retrieval? A case study with broadcast news. *IEEE Trans. Multimed.* **9**(5), 958–966 (2007)
 38. Hunter, J., Drennan, J., Little, S.: Realizing the hydrogen economy through Semantic Web technologies. *IEEE Intell. Syst.* **19**(1), 40–47 (2004)
 39. Little, S., Hunter, J.: Rules-by-example – a novel approach to semantic indexing and querying of images. In: Proceedings of the Third International Semantic Web Conference (ISWC 2004), Hiroshima. Lecture Notes in Computer Science, vol. 3298, pp. 534–548. Springer, Berlin (2004)
 40. Hollink, L., Little, S., Hunter, J.: Evaluating the application of semantic inferencing rules to image annotation. In: International Conference on Knowledge Capture (K-CAP 2005), Banff, pp. 91–98 (2005)
 41. Petridis, K., Anastasopoulos, D., Saathoff, C., Timmermann, N., Kompatsiaris, Y., Staab, S.: M-OntoMat-Annotizer: image annotation, linking ontologies and multimedia low-level features. In: Proceedings of the 10th International Conference on Knowledge-Based Intelligent Information and Engineering Systems, Part 3 (KES 2006), Bournemouth. Lecture Notes in Computer Science, vol. 4253, pp. 633–640. Springer, Berlin (2006)
 42. Maillot, N., Thonnat, M., Boucher, A.: Towards ontology based cognitive vision. In: Proceedings of the International Conference on Computer Vision Systems (ICVS 2003), Graz, pp. 44–53 (2003)
 43. Hudelot, C., Maillot, N., Thonnat, M.: Symbol grounding for semantic image interpretation: from image data to semantics. In: Proceedings of 10th International Conference on Computer Vision Workshops (ICCV 2005), Beijing (2005)
 44. Maillot, N., Thonnat, M.: A weakly supervised approach for semantic image indexing and retrieval. In: Proceedings of the Fourth International Conference on Image and Video Retrieval (CIVR 2005), Singapore. Lecture Notes in Computer Science, vol. 3568, pp. 629–638. Springer, Berlin (2005)
 45. Dasiopoulou, S., Mezaris, V., Kompatsiaris, I., Papastathis, V., Strintzis, M.: Knowledge-assisted semantic video object detection. *IEEE Trans. Circuits Syst. Video Technol.* **15**(10), 1210–1224 (2005)
 46. Neumann, B., Weiss, T.: Navigating through logic-based scene models for high-level scene interpretations. In: Proceedings of the Third International Conference on Computer Vision Systems (ICVS 2003), Graz. Lecture Notes in Computer Science, vol. 2626, pp. 212–222. Springer, Berlin (2003)
 47. Moller, R., Neumann, B., Wessel, M.: Towards computer vision with description logics: some recent progress. In: Workshop on Integration of Speech and Image Understanding, Corfu, pp. 101–115 (1999)
 48. Neumann, B., Mäoller, R.: On scene interpretation with description logics. *Image Vision Comput. Arch.* **26**(1), 247–275 (2008). <http://dx.doi.org/10.1016/j.imavis.2007.08.013>
 49. Hotz, L., Neumann, B., Terzic, K.: High-level expectations for low-level image processing. In: Proceedings of the 31st Annual German

- Conference on AI (KI 2008), Kaiserslautern, pp. 87–94 (2008)
50. Neumann, B.: Bayesian compositional hierarchies – a probabilistic structure for scene interpretation. Technical report FBI-HH-B-282/08. Department of Informatics, Hamburg University (2008)
 51. Peraldi, I.E., Kaya, A., Melzer, S., Möller, R., Wessel, M.: Towards a media interpretation framework for the semantic web. In: Proceedings of the International Conference on Web Intelligence (WI 2007), Silicon Valley, pp. 374–380 (2007)
 52. Dubois, D., Prade, H.: Possibility theory, probability theory and multiple-valued logics: a clarification. *Ann. Math. Artif. Intell.* **32**(1–4), 35–66 (2001)
 53. Sciascio, E.D., Donini, F.: Description logics for image recognition: a preliminary proposal. In: International Workshop on Description Logics (DL 1999), Linköping (1999)
 54. Sciascio, E.D., Donini, F., Mongiello, M.: Structured knowledge representation for image retrieval. *J. Artif. Intell. Res.* **16**, 209–257 (2002)
 55. Dasiopoulou, S., Kompatsiaris, I., Strintzis, M.: Using fuzzy DLs to enhance semantic image analysis. In: Proceedings of Third International Conference on Semantic and Digital Media Technologies (SAMT 2008), Koblenz, pp. 31–46 (2008)
 56. Dasiopoulou, S., Kompatsiaris, I., Strintzis, M.: Applying fuzzy DLs in the extraction of image semantics. *J. Data Semant.* **14**, 105–132 (2009)
 57. Simou, N., Athanasiadis, T., Stoilos, G., Kollias, S.: Image indexing and retrieval using expressive fuzzy description logics. *Signal Image Video Process.* **2**(4), 321–335 (2008)
 58. Hudelot, C., Atif, J., Bloch, I.: Fuzzy spatial relation ontology for image interpretation. *Fuzzy Sets Syst.* **159**(15), 1929–1951 (2008)
 59. Straccia, U.: Reasoning within fuzzy description logics. *J. Artif. Intell. Res.* **14**, 137–166 (2001)
 60. Stoilos, G., Stamou, G., Tzouvaras, V., Pan, J., Horrocks, I.: The fuzzy description logic f-SHIN. In: Proceedings of the International Workshop on Uncertainty Reasoning for the Semantic Web (URSW 2005), Galway, pp. 67–76 (2005)
 61. Ding, Z.: Bayesowl: a probabilistic framework for semantic web. Ph.D. thesis, University of Maryland, Baltimore County (2005)
 62. da Costa, P., Laskey, K., Laskey, K.: PR-OWL: a Bayesian ontology language for the semantic web. In: Proceedings of the Fourth International Workshop on Uncertainty Reasoning for the Semantic Web (URSW 2008), Karlsruhe. Lecture Notes in Computer Science, vol. 5327, pp. 88–107. Springer, Berlin (2008)
 63. Town, C.: Ontological inference for image and video analysis. *Mach. Vis. Appl.* **17**(2), 94–115 (2006)
 64. Tran, S., Davis, L.: Event modeling and recognition using Markov logic networks. In: Proceedings of the 10th European Conference on Computer Vision, Part II (ECCV 2008), Marseille, pp. 610–623 (2008)
 65. Richardson, M., Domingos, P.: Markov logic networks. *Mach. Learn.* **62**(1–2), 107–136 (2006)
 66. Francois, A., Nevatia, R., Hobbs, J., Bolles, R.: Verl: an ontology framework for representing and annotating video events. *IEEE Multimed.* **12**(4), 76–86 (2005)
 67. Dasiopoulou, S., Kompatsiaris, I.: Trends and issues in description logics frameworks for image interpretation. In: Proceedings of the Sixth Hellenic Conference on Artificial Intelligence: Theories, Models and Applications (SETN 2010), Athens, pp. 61–70 (2010)
 68. Schopman, B., Brickley, D., Aroyo, L., van Aart, C., Buser, V., Siebes, R., Nixon, L., Miller, L., Malaise, V., Minno, M., Mostarda, M., Palmisano, D., Raimond, Y.: NoTube: making the web part of personalised TV. In: Proceedings of the WebSci10: Extending the Frontiers of Society Online, Raleigh (2010)
 69. Taylor, A.: Introduction to Cataloging and Classification. Library and Information Science Text Series. Libraries Unlimited, Santa Barbara (2006)
 70. Sowa, J.: Knowledge Representation. Logical, Philosophical, and Computational Foundations. Brooks/Cole, Pacific Grove (2000)
 71. Doerr, M.: The CIDOC CRM – an ontological approach to semantic interoperability of metadata. *AI Mag.* **24**(3), 75–92 (2003)
 72. Ruotsalo, T., Hyvönen, E.: An event-based method for making heterogeneous metadata schemas and annotations semantically interoperable. In: Proceedings of the Sixth International Semantic Web Conference, Second Asian Semantic Web Conference (ISWC 2007 + ASWC 2007),

- Busan. *Lecture Notes in Computer Science*, vol. 4825, pp. 409–422. Springer, Berlin (2007)
73. Borgo, S., Masolo, C.: Foundational choices in DOLCE. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*. *International Handbooks on Information Systems*, 2nd edn., pp. 403–421. Springer, Dordrecht (2009)
 74. Fellbaum, C. (ed.): *WordNet. An Electronic Lexical Database*. MIT Press, Cambridge (2001)
 75. Yano, K., Nakaya, T., Isoda, Y., Takase, Y., Kawasumi, T., Matsuoka, K., Seto, T., Kaawahara, D., Tsukamoto, A., Inoue, M., Kirimura, T.: Virtual Kyoto: 4D GIS comprising spatial and temporal dimensions. *J. Geogr.* **117**(2), 464–476 (2008)
 76. Kauppinen, T., Väättä, J., Hyvönen, E.: Creating and using geospatial ontology time series in a semantic cultural heritage portal. In: *Proceedings of the Fifth European Semantic Web Conference (ESWC 2008)*, Tenerife. *Lecture Notes in Computer Science*, vol. 5021, pp. 110–123. Springer, Berlin (2008)
 77. Nagypál, G., Motik, B.: A fuzzy model for representing uncertain, subjective, and vague temporal knowledge in ontologies. In: *On the Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE – OTM Confederated International Conferences (CoopIS, DOA, and ODBASE 2003)*, Catania, pp. 906–923 (2003)
 78. Kauppinen, T., Mantegari, G., Paakkari, P., Kuittinen, H., Hyvönen, E., Bandini, S.: Determining relevance of imprecise temporal intervals for cultural heritage information retrieval. *Int. J. Hum. Comput. Stud.* **86**(9), 549–560 (2010). Elsevier
 79. Allen, J.F.: Maintaining knowledge about temporal intervals. *Commun. ACM* **26**(11), 832–843 (1983)
 80. Hyvönen, E., Mäkelä, E., Salminen, M., Valo, A., Viljanen, K., Saarela, S., Junnila, M., Kettula, S.: MuseumFinland – Finnish museums on the semantic web. *J. Web Semant.* **3**(2), 224–241 (2005)
 81. McCarty, W.: *Humanities Computing*. Palgrave Macmillan, Basingstoke (2005)
 82. Sheth, A., Aleman-Meza, B., Arpinar, I.B., Bertram, C., Warke, Y., Ramakrishnan, C., Halaschek, C., Anyanwu, K., Avant, D., Arpinar, F.S., Kochut, K.: Semantic association identification and knowledge discovery for national security applications. *J. Database Manag. Database Technol.* **16**(1), 33–53 (2005)
 83. Hyvönen, E., Mäkelä, E., Kauppinen, T., Alm, O., Kurki, J., Ruotsalo, T., Seppälä, K., Takala, J., Puputti, K., Kuittinen, H., Viljanen, K., Tuominen, J., Palonen, T., Frosterus, M., Sinkkilä, R., Paakkari, P., Laitio, J., Nyberg, K.: CultureSampo – Finnish culture on the semantic web 2.0. In: *Proceedings of the Museums and the Web (MW 2009)*, Indianapolis (2009)
 84. Schreiber, G., Amin, A., Aroyo, L., van Assen, M., de Boer, V., Hardman, L., Hildebrand, M., Omelayenko, B., van Ossenbruggen, J., Tordai, A., Wielemaker, J., Wielinga, B.J.: Semantic annotation and search of cultural-heritage collections: The MultimediaN E-Culture demonstrator. *J. Web Semant.* **6**(4), 243–249 (2008)
 85. Junnila, M., Hyvönen, E., Salminen, M.: Describing and linking cultural semantic content by using situations and actions. In: Robering, K. (ed.) *Information Technology for the Virtual Museum*. LIT verlag, Berlin (2008)
 86. Byrne, K.: *Populating the Semantic Web – Combining text and relational databases as RDF graphs*. Ph.D. thesis, University of Edinburgh, Supp. 32 (2009)
 87. Hyvönen, E.: Semantic portals for cultural heritage. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*, 2nd edn., pp. 757–778. Springer, Dordrecht (2009)
 88. van Hage, W.R., Stash, N., Wang, Y., Aroyo, L.: Finding your way through the Rijksmuseum with an adaptive mobile museum guide. In: *The Semantic Web: Research and Applications, Seventh Extended Semantic Web Conference (ESWC 2010)*, Proceedings, Part I, Heraklion. *Lecture Notes in Computer Science*, vol. 6088, pp. 46–59. Springer, Berlin (2010)
 89. Becker, C., Bizer, C.: DBpedia mobile: a location-enabled linked data browser. In: *Proceedings of the First Workshop about Linked Data on the Web (LDOW 2008)*, Beijing (2008); Kobilarov, G., Scott, T., Raimond, Y., Oliver, S., Sizemore, C., Smethurst, M., Bizer, C., Lee, R.: Media meets semantic web – how the BBC uses DBpedia and linked data to make connections. In: *The Semantic Web: Research and Applications, Proceedings of the Seventh Extended Semantic Web Conference (ESWC 2010)*, Part I,

- Heraklion. Lecture Notes in Computer Science, vol. 6088, pp. 723–737. Springer, Berlin (2010)
90. Euzenat, J., Shvaiko, P.: *Ontology Matching*. Springer, Berlin (2007)
 91. Hyvönen, E., Viljanen, K., Tuominen, J., Seppälä, K.: Building a national semantic web ontology and ontology service infrastructure – the FinnONTO approach. In: *Proceedings of the Fifth European Semantic Web Conference (ESWC 2008)*, Tenerife. Lecture Notes in Computer Science, vol. 5021, pp. 95–109. Springer, Berlin (2008)
 92. Rui, Y., Huang, T., Chang, S.: Image retrieval: current techniques, promising directions and open issues. *J. Vis. Commun. Image Represent.* **10**(4), 39–62 (1999)
 93. Lew, M.S., Sebe, N., Djeraba, C., Jain, R.: Content-based multimedia information retrieval: state of the art and challenges. *ACM Trans. Multimed. Comput. Commun. Appl.* **2**, 1–19 (2006)
 94. Hollink, L.: *Semantic annotation for retrieval of visual resources*. Ph.D. thesis, Free University of Amsterdam. SIKS Dissertation Series, No. 2006-24 (2006)
 95. Pollitt, A.S.: The key role of classification and indexing in view-based searching. Technical Report. University of Huddersfield, UK. <http://www.ifla.org/IV/ifla63/63polst.pdf> (1998)
 96. Hearst, M., Elliott, A., English, J., Sinha, R., Swearingen, K., Lee, K.-P.: Finding the flow in Web site search. *Commun. ACM* **45**(9), 42–49 (2002)
 97. Hyvönen, E., Saarela, S., Viljanen, K.: Application of ontology techniques to view-based semantic search and browsing. In: *The Semantic Web: Research and Applications. Proceedings of the First European Semantic Web Symposium (ESWS 2004)*, Heraklion. Lecture Notes in Computer Science, vol. 3053, pp. 92–106. Springer, Berlin (2004)
 98. Sacco, G.M.: Dynamic taxonomies: guided interactive diagnostic assistance. In: Wickramasinghe, N. (ed.) *Encyclopedia of Healthcare Information Systems*. Idea Group, Hershey (2005)
 99. Hildebrand, M., van Ossenbruggen, J., Hardman, L.: /facet: a browser for heterogeneous Semantic Web repositories. In: *Proceeding of the Fifth International Semantic Web Conference (ISWC 2006)*, Athens, GA. Lecture Notes in Computer Science, vol. 4273, pp. 272–285. Springer, Berlin (2006)
 100. Holí, M.: *Crisp, fuzzy, and probabilistic faceted semantic search*. Dissertation, School of Science and Technology, Aalto University, Espoo (2010)
 101. English, J., Hearst, M., Sinha, R., Swearingen, K., Lee, K.-P.: *Flexible search and navigation using faceted metadata*. Technical report. School of Information Management and Systems, University of Berkeley, Berkeley (2003)
 102. Hyvönen, E., Mäkelä, E.: Semantic autocompletion. In: *Proceedings of the First Asian Semantic Web Conference (ASWC 2006)*, Beijing. Lecture Notes in Computer Science, vol. 4185, pp. 739–751. Springer, Heidelberg (2006)
 103. Hildebrand, M., van Ossenbruggen, J.R.: Configuring semantic web interfaces by data mapping. In: *Proceedings of the Workshop on: Visual Interfaces to the Social and the Semantic Web (VISSW 2009)*, Sanibel Island (2009)
 104. Burke, R.: Knowledge-based recommender systems. In: Kent, A. (ed.) *Encyclopedia of Library and Information Systems*, vol. 69. Marcel Dekker, New York (2000)
 105. Geroimenko, V., Chen, C. (eds.): *Visualizing the Semantic Web: XML-Based Internet and Information Visualization*. Springer, Berlin (2002)
 106. Kauppinen, T., Henriksson, R., Väätäinen, J., Deichstetter, C., Hyvönen, E.: Ontology based modeling and visualization of cultural spatio-temporal knowledge. In: *Semantic Web at Work – Proceedings of STeP 2006*, Finnish AI Society, Espoo (2006)
 107. Adomavicius, G.: Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.* **17**(6), 734–749 (2005)
 108. Viljanen, K., Käsälä, T., Hyvönen, E., Mäkelä, E.: ONTODELLA – a projection and linking service for Semantic Web applications. In: *Proceedings of the 17th International Conference on Database and Expert Systems Applications (DEXA 2006)*, Krakow (2006)
 109. Ruotsalo, T., Mäkelä, E., Kauppinen, T., Hyvönen, E., Haav, K., Rantala, V., Frosterus, M., Dokoohaki, N., Matskin, M.: Smartmuseum: personalized context-aware access to digital cultural heritage. In: *Proceedings of the International Conferences on Digital Libraries and the Semantic Web 2009 (ICSD 2009)*, Trento (2009)

110. Wang, Y., Stash, N., Aroyo, L., Gorgels, P., Rutledge, L., Schreiber, G.: Recommendations based on semantically-enriched museum collection. *J. Web Semant.* **6**(4), 283–290 (2008)
111. Spagnuolo, M., Falcidieno, B.: 3D media and the semantic web. *IEEE Intell. Syst.* **24**(2), 90–96 (2009)
112. State of the art report on 3D content in archaeology and cultural heritage. FOCUS K3D deliverable 2.4.1 (2009)
113. Hauswirth, M., Decker, S.: Semantic reality – Connecting the real and the virtual world. In: Position Paper at Microsoft SemGrail Workshop, Redmond (2007)
114. Dasiopoulou, S., Tzouvaras, V., Kompatsiaris, I., Strintzis, M.G.: Enquiring MPEG-7 based ontologies. *Multimed. Tools Appl.* **46**(2), 331–370 (2010)
115. Staab, S., Studer, R. (eds.): *Handbook on Ontologies*. Springer, Dordrecht (2004)
116. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F.: *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, Cambridge (2003)
117. Bertini, M., Del Bimbo, A., Serra, G., Torniai, C., Cucchiara, R., Grana, C., Vezzani, R.: Dynamic pictorially enriched ontologies for digital video libraries. *IEEE Multimed.* **16**(2), 42–51 (2009)
118. Shet, V.D., Neumann, J., Ramesh, V., Davis, L.S.: Bilattice-based logical reasoning for human detection. In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2007)*, Minneapolis, pp. 1–8 (2007)
119. Patkos, T., Chrysakis, I., Bikakis, A., Plexousakis, D., Antoniou, G.: A reasoning framework for ambient intelligence. In: *Proceedings of the Sixth Hellenic Conference on Artificial Intelligence: Theories, Models and Applications (SETN 2010)*, Athens, pp. 213–222 (2010)
120. Kowalski, R.A., Sergot, M.J.: A logic-based calculus of events. In: *Foundations of Knowledge Base Management*, pp. 23–55. Springer, Berlin (1985). ISBN 3-540-18987-4
121. Biancalana, C., Micarelli, A., Squarcella, C.: Nereau: a social approach to query expansion. In: *Proceeding of the 10th ACM International Workshop on Web Information and Data Management (WIDM 2008)*, Napa Valley, pp. 95–102. ACM, New York (2008)
122. Brusilovsky, P., Maybury, M.T.: From adaptive hypermedia to the adaptive web. *Commun. ACM* **45**(5), 30–33 (2002)
123. Carmagnola, F., Cena, F., Gena, C., Torre, I.: A semantic framework for adaptive web-based systems. In: Bouquet, P., Tummarello, G. (eds.) *Semantic Web Applications and Perspectives (SWAP 2005)*, *Proceedings of the Second Italian Semantic Web Workshop, Trento*. CEUR Workshop Proceedings, vol. 166. CEUR-WS.org (2005)
124. Ginsberg, M.L.: Multivalued logics: a uniform approach to reasoning in artificial intelligence. *Comput. Intell.* **4**, 265–316 (1988)



22 Semantic Web Services

Carlos Pedrinaci¹ · John Domingue¹ · Amit P. Sheth²

¹The Open University, Milton Keynes, UK

²Wright State University, Dayton, Ohio, USA

22.1	<i>Introduction</i>	978
22.2	<i>Scientific Overview</i>	980
22.2.1	Conceptual Models	984
22.2.1.1	Top-Down Approaches	984
22.2.1.2	Bottom-Up Approaches	991
22.2.2	Semantic Web Services Infrastructure	1000
22.2.2.1	OWL-S Tools and Engines	1000
22.2.2.2	WSMX	1003
22.2.2.3	IRS-III	1007
22.2.2.4	METEOR-S	1012
22.3	<i>Example Applications</i>	1017
22.3.1	Applying Semantic Web Services in eGovernment	1018
22.3.1.1	Change of Circumstances	1019
22.3.1.2	eMerges	1022
22.4	<i>Related Resources and Key Papers</i>	1026
22.4.1	Key Papers	1026
22.4.2	Related Papers	1029
22.5	<i>Future Issues</i>	1030
22.6	<i>Cross-References</i>	1032

Abstract: In recent years, service-orientation has increasingly been adopted as one of the main approaches for developing complex distributed systems from reusable components called services. Realizing the potential benefits of this software engineering approach requires semiautomated and automated techniques as well as tools for searching or locating services, selecting the suitable ones, composing them into complex processes, resolving heterogeneity issues through process and data mediation, and reducing other tedious yet recurrent tasks with minimal manual effort. Just as semantics has brought significant benefits to search, integration, and analysis of data, it is also seen as a key to achieving a greater level of automation to service-orientation. This has led to research and development, as well as standardization efforts on Semantic Web Services. Activities related to Semantic Web Services have involved developing conceptual models or ontologies, algorithms, and engines that could support machines in semiautomatically or automatically discovering, selecting, composing, orchestrating, mediating, and executing services. This chapter provides an overview of the area after nearly a decade of research. The chapter presents the main principles and conceptual models proposed thus far, including OWL-S, Web Service Modeling Ontology (WSMO), and Semantic Annotations for WSDL (SAWSDL)/Managing End-to-End Operations-Semantics (METEOR-S), as well as recent approaches that provide lighter solutions and bring support for the increasingly popular Web APIs and RESTful services, like SA-REST, WSMO-Lite, and MicroWSMO. The chapter also describes the main engines and frameworks developed by the research community, including discovery engines, composition engines, and even integrated frameworks that are able to use these semantic descriptions of services to support some of the typical activities related to services and service-based applications. Next, the ideas and techniques described are illustrated through two use cases that integrate Semantic Web Services technologies within real-world applications. Finally, a set of key resources that would allow the reader to reach a greater understanding of the field is provided, and the main issues that will drive the future of Semantic Web Services (SWS) are outlined.

22.1 Introduction

Service-orientation is an approach to developing software by combining reusable and possibly distributed components called services [1]. Since it was first proposed, there has been much research and development around service-orientation principles, architectural models, languages providing means for describing components as reusable services, languages for defining service compositions, and a plethora of software for supporting this vision have been implemented.

Although each major vendor promotes a distinct set of concrete technologies and solutions, all these approaches have in common the identification of a services registry where existing services can be located, and the provisioning of means (e.g., languages, tools, and engines) to compose these services to achieve more complex functionalities, therefore supporting the seamless creation of complex distributed solutions out of

preexisting components [2]. This is essentially an architectural model for developing software out of reusable and distributed services, which is often referred to as Service-Oriented Architecture (SOA).

Although technology independent, SOA is typically implemented using Web Service technologies such as the Web Service Description Language (WSDL) and SOAP [2]. WSDL was created specifically for this purpose, providing several useful constructs for describing services, including operations to describe service methods, parameter descriptions via XML Schema, and information about the type of protocol needed to invoke the service (e.g., SOAP over HTTP). Many activities require additional functionalities not captured in the basic service specification supported by WSDL, and the use of multiple services to accomplish them. Correspondingly, the so-called WS-* standards and composition languages [3] have been defined in order to provide further capabilities to service-oriented solutions [1, 2].

SOA is commonly lauded as a silver bullet for Enterprise Application Integration, implementation of interorganizational business processes, and even as a general solution for the development of all complex distributed applications. However, their uptake on a web-scale has been significantly less prominent than initially anticipated [4]. Instead, more recently, plain and simple Web technologies (HTTP, XML, and JSON), which underlie machine-oriented Web applications and APIs, are increasingly being used to provide added-value solutions that combine information from diverse sources seamlessly, constituting simple and “lightweight” service-oriented software. These recent services are commonly referred to as RESTful services – when they follow REST principles [5] – or Web APIs in general.

Independently from the technologies adopted and despite the appealing characteristics of service-orientation principles and technologies, the systematic development of service-oriented applications remains limited and effectively one is still far from truly benefiting from the promised simplicity for constructing agile and interoperable systems. The fundamental reason for this lies on the need for software developers to devote significant labor to discovering sets of suitable services, interpreting them, developing software that overcomes their inherent data and process mismatches, and finally combining them into a complex composite process.

Semantic Web Services (SWS) were proposed in order to pursue the vision of the Semantic Web presented in [6], whereby intelligent agents would be able to exploit semantic descriptions in order to carry out complex tasks on behalf of humans [7]. This early work on SWS was the meeting point between Semantic Web, Agents, and Web Services technologies. Gradually, however, research focused more prominently on combining Web Services with Semantic Web technologies in order to better support the discovery, composition, and execution of Web Services, leaving aspects such as systems’ autonomy, more typical of agent-based systems, somewhat aside.

Research on SWS has been active and fruitful over the years leading to a number of conceptual models, representation languages, as well as to a plethora of software components and even integrated execution frameworks that cover diverse tasks within the life cycle of Web Services and service-oriented applications. This chapter aims at providing an overview of the

main results achieved so far, giving the reader pointers for gathering further insights and details on how these solutions have been devised. SWS research builds upon results and techniques from a wide range of fields and therefore, for the sake of clarity and space, the focus is on the main approaches and techniques directly applied to SWS proposed thus far, leaving the more systematic review of related fields to the interested reader.

The remainder of this chapter is organized as follows. First, the types of semantics present in a service and the main concepts surrounding SWS are covered briefly so that the reader can understand better the subsequent sections. Next, the main conceptual models for describing SWS devised so far are presented. On the basis of these conceptual models, the main frameworks and components that exploit SWS to achieve concrete tasks are introduced. After the technical details have been introduced, a couple of example applications are presented that aim at illustrating some of the concepts exposed in earlier sections of this chapter. Finally, a set of key references and related papers that are considered of particular relevance are included, and the chapter concludes introducing future issues that are expected to be at the center of research on SWS in the forthcoming years.

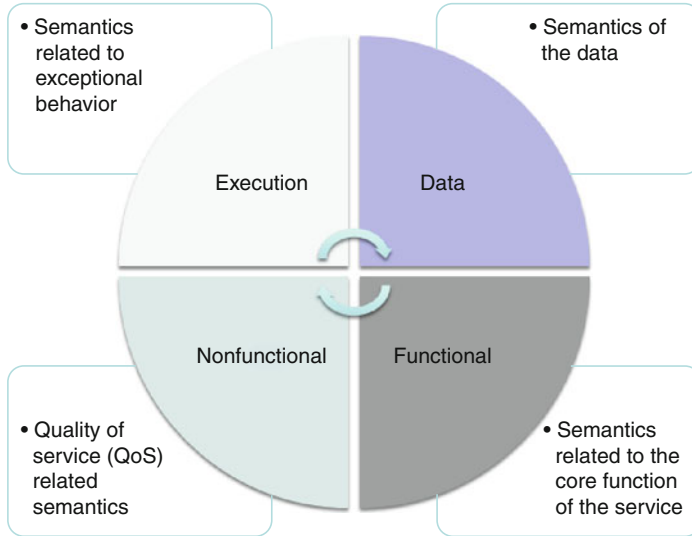
22.2 Scientific Overview

Semantic Web Services were first proposed by McIlraith et al. in [7] as an extension of Web Services with semantic descriptions in order to provide formal declarative definitions of their interfaces as well as to capture declaratively what the services do. Early on, Sheth et al. [8, 9] introduced the four main types of semantics (see [Fig. 22.1](#)), that corresponding semantic descriptions can capture:

1. Data semantics: The semantics pertaining to the data used and exposed by the service
2. Functional semantics: Semantics pertaining to the functionality of the service
3. Nonfunctional semantics: Semantics related to the nonfunctional aspects of the service, for example, quality of service (QoS), security, or reliability
4. Execution semantics: Semantics related to exceptional behaviors such as runtime errors

The essential characteristic of SWS is therefore the use of languages with well-defined semantics covering the subset of the mentioned categories that are amenable to automated reasoning. Several languages have been used so far including those from the Semantic Web, for example, Resource Description Framework (RDF(S)) and Web Ontology Language (OWL), SWS-specific languages such as the Web Service Modeling Language (WSML), or others originating from research on Knowledge-Based Systems such as F-Logic and Operational Conceptual Modeling Language (OCML).

On the basis of these semantic descriptions, SWS technologies seek to automate the tasks involved in the life cycle of service-oriented applications, which include the discovery and selection of services, their composition, their execution, and their monitoring among others. Part of the research on SWS has been devoted precisely to identifying the requirements for SWS systems, the tasks involved and even to defining conceptual frameworks and architectures that cover the entire life cycle of SWS [7, 10–13].



■ Fig. 22.1

Four types of semantics introduced in [8, 9]

The different conceptual frameworks proposed make particular emphasis on certain aspects such as better supporting the decoupling and scalability of the solutions, on supporting the interactions between agents, or on reaching a solution that appropriately fulfills humans' expectations. Instead of adopting a concrete framework, it is herein distilled what are typically the essential features of all the frameworks proposed in order to give an overall view on the field and help the reader to better understand the remainder of the chapter. Later the concrete proposals will be covered in more detail.

All the frameworks take as a starting point service-orientation principles and are strongly based on Service-Oriented Architectures. Hence, they view the construction of systems as a process involving the encapsulation of reusable components as services, their publication into shared registries, the location of existing and suitable services from these shared repositories, their composition into executable workflows, and their eventual execution or enactment. SWS frameworks essentially propose the application of semantics to reach a higher level of automation throughout these tasks.

The remainder of this section identifies the main tasks and concepts typically utilized and mentioned within in the SWS field in order to provide a common set of definitions. Some of the papers cited in this chapter may use a slightly different terminology often due to the evolution of the research in the area. Where appropriate, typical uses of different terminology that readers may encounter shall be identified. The definitions exposed herein are largely in line with those provided by the W3C Web Services Glossary [14].

Crawling is the task in charge of browsing the Web in order to locate existing services. This task is essentially identical to any other Web crawling endeavor with the difference that the information sought is not Web pages but rather WSDL files or more recently HTML pages describing Web APIs.

Discovery involves locating services that are able to fulfill certain user requirements. This task encompasses, starting from abstract definitions of users' needs, operationalizing the requirements such that they can be used for identifying Web Services that can subsequently be used for *discovering* whether they can provide the desired service. This definition of discovery, which is based on that given in [15], distinguishes the notion of service from a business perspective (e.g., booking a concrete flight), from the notion of Web Service, which is the functional component able to provide a certain service (e.g., booking flights operated by a particular airline). This understanding of service discovery aims at providing functionality closer to human needs by clearly distinguishing between services that have effect on the real world from the Web Services that can be invoked to achieve these changes. It therefore distinguishes the location of possibly suitable services from the actual discovery of those that can really provide the service sought for; after all, not all the flight booking Web Services will allow booking any flight from any company.

The term discovery is perhaps the most widely applied term in SWS research; however, in the large majority of the cases the task that it is referred to is what is here referred to as service matching or service matchmaking. It is worth noting that the definition just outlined involves but is not limited to service matching and may also require an additional activity called service crawling.

Matching, also referred to as matchmaking in several papers, is the task that given a request for some *kind of Web Service* tries to identify Web Service advertisements that match to a certain degree the request. Research in Web Service matching has devoted substantial efforts to formalizing Web Services functionality in ways that can support automatic matching using reasoners. In general, Web Service functionality is specified in terms of inputs, outputs, preconditions, and effects (IOPEs) (in WSMO, see section "WSMO," assumptions and post-conditions are also considered). The set of known Web Services advertisements are then matched against the functionality specifications sought for using reasoners and additional heuristics. The result is a restricted set of Web Services according to different degrees of matching, which most often contemplate at least exact, plug-in – when the advertisement subsumes the request, subsumes – when the request subsumes the advertisement, and fail.

Ranking is the task that given a set of Web Services obtained from the matching process, ranks the different matches according to a set of preferences. These preferences are usually given at invocation time and are specified in terms of the nonfunctional properties of Web Services, for example, price and quality of service. In this manner, it is possible to order Web Services that are able to provide the required functionality based on other kinds of criteria. It is worth noting in this respect, that the matching degree described earlier is one kind of simple criteria typically applied.

Selection is the task that having obtained a list of (possibly ranked) suitable Web Services, selects one to be invoked. This task is often performed by humans but there also exist systems that carry it out automatically based on prior ranking criteria. Since in most cases there may exist data heterogeneities, most of the systems implementing automated selection also provide data mediation facilities so that invocation can take place.

Composition is the task in charge of combining Web Services in order to achieve a complex task. Typically, this task is triggered whenever the system is unable to find a Web Service that fulfills all the requirements. The result of a composition task is an orchestration of Web Services that, given some initial conditions and a set of Web Services, would lead to the desired state when executed. Most of the work in automated Semantic Web Service composition has been approached as a planning task, which benefits from the formal specification of Web Services inputs, outputs, preconditions, and effects to generate suitable orchestrations.

Orchestration defines the sequence and conditions for the enactment of Web Services in order to achieve a complex objective by appropriately combining the functionality provided by existing Web Services. Orchestration definitions include data-flow (i.e., how the data are propagated and used throughout the process) and control-flow (i.e., when should a certain activity be executed). There exists a wide range of process specification languages that have been defined over the years. In this chapter, those that have been used in SWS research are introduced. The reader is referred to [16–18] for further insights.

Choreography describes the interactions of services with their users, whereby any client of a Web Service, may it be a human or a machine, is considered a user. A choreography defines the expected behavior of a Web Service, that is, the exchanged messages, from a client's point of view. Choreography interpretation leads to a successful invocation of a Web Service independently from how the execution of the Web Service is performed internally.

Mediation is necessary in environments where having heterogeneous components is frequent. Heterogeneity is one of the main characteristics of the Web and therefore any Web-oriented solution must face this issue in one way or another. Mediation is a principle by which an intermediate element, a mediator, is introduced between two elements to resolve their heterogeneities without having to adapt one or the other. In a nutshell, heterogeneity in Web Services can affect three main aspects: (i) the terminology used; (ii) the representation and network-level protocol used for communication; and (iii) the application-level protocol expected by two interacting parties.

Issues concerning terminology and the representation of data are commonly referred to as data mediation. Data mediation aims at resolving the mismatches between the data handled by both services typically by approaching it as an ontology mapping/transformation problem. Conversely, protocol mediation aims at achieving a successful interaction between two processes (or Web Services) by ensuring that the message exchanges between both processes are as they expect. Protocol mediation, which may not always be resolvable, often involves buffering and the reordering of messages, or combining them so that each process can reach a successful end state.

Invocation is concerned with the actual call to an operation of a Web Service. Invocation is therefore closely related to choreographies in that the latter will specify an order for performing a set of invocations.

Grounding specifies how certain activities are mapped into low-level operations with Web Services. The need for grounding specifications comes from the fact that Semantic Web Services are essentially handled at the semantic level where invocation details are often disregarded. In most of the approaches, grounding definitions are basically pointers to existing operation definitions.

Lifting refers to the transformation of information from its representation at the syntactic level used by the Web Service (typically XML) into its semantic counterpart. Lifting is usually expressed declaratively so that it can directly be interpreted by some engine in order to transform the data into some semantic representation, for example, RDF, OWL, WSMO. Given that many services exchange XML data, Extensible Stylesheet Language Transformations (XSLT) is often the language of choice.

Lowering refers to the task that takes information represented semantically and transforms it into some syntactic representation that can be used for communicating with the Web Service. This task is the inverse of Lifting, and likewise is often approached with XSLT.

22.2.1 Conceptual Models

Central to the work on SWS are the semantic descriptions or annotations of services that support automating to a greater extent tasks such as their discovery, selection, and composition. Consequently, much effort has been devoted over the years to devising conceptual models able to support creating suitable semantic descriptions for services. In the remainder of this section the main approaches, which have been divided into top-down approaches and bottom-up approaches are introduced. Top-down approaches to the development of Semantic Web Services like the Web Service Modeling Ontology (WSMO) [19, 20] and OWL-S [21], are based on the definition of high-level ontologies providing expressive frameworks for describing Web Services. On the other hand, bottom-up models, for example, WSDL-S [22] and SAWSDL [23], adopted an incremental approach to attaching semantics to existing Web Services standards by adding specific extensions that connect the syntactic definitions to their semantic annotations.

22.2.1.1 Top-Down Approaches

OWL-S

OWL-S [21], formerly DAML-S, is an ontology for the description of Semantic Web Services expressed in OWL [24]. OWL-S, which was submitted to W3C in 2004, defines an upper ontology for semantically describing Web Services along three main aspects:

- The *Service Profile* describes *what the service does* in terms of inputs, outputs, preconditions, and effects (IOPEs).
- The *Service Model* describes *how a service works* in terms of a process model that may describe a complex behavior over underlying services.
- The *Service Grounding* describes *how the service can be accessed*, usually by grounding to WSDL.

The Service Profile provides the core functional description of services used for advertising. This description provides a high-level representation on what the service does in a manner that is suitable for software agents to discover whether a service is adequate for

their purposes or not. A service is described mainly in terms of its functional parameters: inputs, outputs, preconditions, and effects (IOPEs), see [Fig. 22.2](#). Basically, inputs and outputs specify semantically the kinds of parameters handled by the service. Preconditions are logical expressions that specify the conditions that are required for the service to be executed successfully (e.g., the customer has to be located in the USA). Effects, also referred to as Results in OWL-S, on the other hand, specify changes in the world should the execution of the service be successful (e.g., the book is shipped to the given address). Additionally, the Service Profile includes support for capturing information such as classifications with respect to reference taxonomies, the name of the service, and textual descriptions.

The Service Model informs clients about how to use the service. It does so by specifying the semantic content of requests, replies, the conditions under which certain results hold, and how clients have to invoke the service. In order to tell clients how to interact with a service, OWL-S views services as processes with a set of inputs, outputs, preconditions, and effects, as well as a process model specifying the ways in which a client may interact with the service. Reasoning support for OWL-S is provided primarily by OWL-DL reasoners. However, OWL-DL is often not sufficiently expressive or not suitable for defining preconditions and effects. For these cases, OWL-S supports the specification

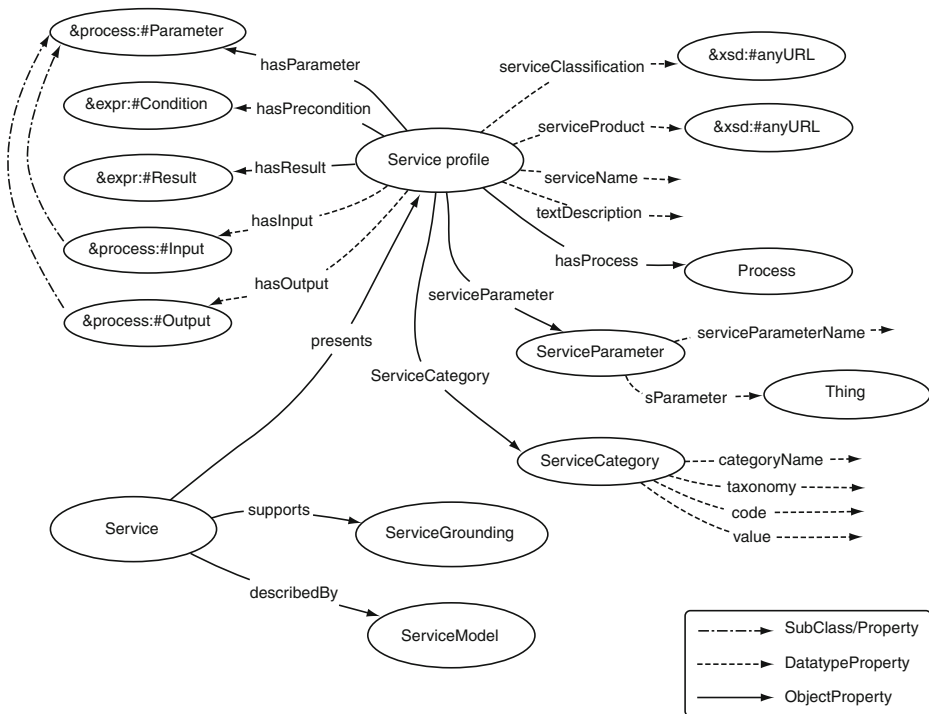


Fig. 22.2

The OWL-S ontology (Figure adapted from [21])

of preconditions and effects by means of literals, either string literals or XML literals, which allow modelers to adopt arbitrary languages, such as SWRL [25], for including expressions that are beyond the expressivity of OWL.

OWL-S provides an advanced solution for conditioning outputs and effects so that one can express that a certain effect in the world would only occur if a certain condition held for the inputs (e.g., you get a voucher as a present if you spend more than a certain amount). Additionally, OWL-S includes additional modeling constructs in order to provide means for expressing complex processes. OWL-S distinguishes three main kinds of processes: *Atomic Processes*, *Composite Processes*, and *Simple Processes*. See [Listing 22.1](#) for an example of an Atomic Process.

Listing 22.1: Example of an OWL-S atomic process (Taken from [17])

```
<process:AtomicProcess rdf:ID="Purchase">
  <process:hasInput>
    <process:Input rdf:ID="ObjectPurchased"/>
  </process:hasInput>
  <process:hasInput>
    <process:Input rdf:ID="PurchaseAmt"/>
  </process:hasInput>
  <process:hasInput>
    <process:Input rdf:ID="CreditCard"/>
  </process:hasInput>
  <process:hasOutput>
    <process:Output rdf:ID="ConfirmationNum"/>
  </process:hasOutput>
  <process:hasResult>
    <process:Result>
      <process:hasResultVar>
        <process:ResultVar rdf:ID="CreditLimH">
          <process:parameterType rdf:resource="&ecom;
            #Dollars"/>
        </process:ResultVar>
      </process:hasResultVar>
    </process:Result>
  </process:hasResult>
  <process:inCondition>
    <expr:KIF-Condition>
      <expr:expressionBody>
        (and (current-value (credit-limit ?CreditCard)
          ?CreditLimH)
          (>= ?CreditLimH ?purchaseAmt))
      </expr:expressionBody>
    </expr:KIF-Condition>
  </process:inCondition>
</process:AtomicProcess>
```

```

<process:withOutput>
  <process:OutputBinding>
    <process:toParam rdf:resource="#ConfirmationNum"/>
    <process:valueFunction rdf:parseType="Literal">
      <cc:ConfirmationNum xsd:datatype="xsd;
        #string"/>
    </process:valueFunction>
  </process:OutputBinding>
</process:withOutput>
<process:hasEffect>
  <expr:KIF-Condition>
    <expr:expressionBody>
      (and (confirmed (purchase ?purchaseAmt)
        ?ConfirmationNum)
        (own ?objectPurchased)
        (decrease (credit-limit ?CreditCard)
          ?purchaseAmt))
    </expr:expressionBody>
  </expr:KIF-Condition>
</process:hasEffect>
</process:Result>
...
</process:hasResult>
</process:AtomicProcess>

```

Atomic Processes are processes that are directly invocable and, as far as the service requester is concerned, they only require one single interaction. Atomic Processes therefore require a grounding indicating how invocation messages have to be constructed and the result messages parsed. More details about the grounding of Atomic Processes are provided later.

Composite Processes are processes that require several steps in the interaction and/or multi-server actions. They are also decomposable into other processes (composite or not). In order to support the definition of Composite Processes, OWL-S provides a set of block-oriented control constructs, such as *Sequence*, *If-Then-Else*, or *Repeat While*. It is worth noting, however, that these control-flow definitions do not specify how the service will behave but rather what clients invoking the service could do.

Finally, Simple Processes provide an abstraction mechanism able to provide multiple views over existing Atomic and Composite Processes. Simple Processes are not directly invocable (they are not associated with a grounding), although they have single-step interactions much like Atomic Processes. Simple Processes are mostly used as suitable abstractions for simplifying tasks such as planning and reasoning. However, in order to support the eventual invocation of these processes, OWL-S provides the *realizedBy* and *expandsTo* relations, connecting them to *Atomic Processes* and *Composite Processes*, respectively.

The Service Grounding provides the details necessary for invoking the service. It is therefore concerned with aspects such as the protocol to be used, the message format, their serialization, the transport protocol, and the address of the endpoint to be invoked. In a nutshell, the Service Grounding is a mapping between the parameters handled by Atomic Processes and the messages that carry those parameters in some specific transmittable format. OWL-S does not predefine the language to be used for grounding; however, due to its wide adoption a reference grounding implementation is provided for WSDL.

WSMO

WSMO [19, 20] is a member submission to W3C of an ontology that aims at describing all relevant aspects for the partial or complete automation of discovery, selection, composition, mediation, execution, and monitoring of Web Services. WSMO has its roots in the Web Service Modeling Framework (WSMF) [10] and in Problem-Solving Methods [26], notably the Unified Problem-Solving Method Development Language (UPML) [27], which have been extended and adapted in order to support the automation of the aforementioned tasks for manipulating Web Services.

WSMF provides an overall framework for describing Web Services based on two essential principles for semantic descriptions of Web Services:

- *Strict decoupling*: The various components that realize an application are described in a unitary fashion without regard to the neighboring components. This enables components to be easily linked and promotes scalability following the open and distributed nature of the Web.
- *Centrality of mediation*: Building on the concept of bridges within the UPML framework, WSMF includes the notion of mediators to deal with mismatches that may occur between components. Heterogeneity can occur in terms of data, underlying ontology, protocol, or process. WSMF recognizes the importance of mediation for the successful deployment of Web Services by making mediation a first-class component. A mediator provides a link to a mechanism that can resolve the identified mismatch.

WSMO provides an ontology for describing Web Services based upon WSMF. WSMO identifies four top-level elements as the main concepts, namely *Ontologies*, *Web Services*, *Goals*, and *Mediators* (in this chapter these terms will be used in capitals whenever WSMO elements are referred to). *Ontologies* provide the formal semantics for the terminology used within all other WSMO components. Essentially, WSMO establishes that all resource descriptions and all data interchanged during service usage should be semantically described on the basis of ontologies. Web Services are computational entities that provide some value in a given domain. Goals represent clients' perspective by supporting the representation of users' desires for certain functionality. Finally, Mediators represent elements that handle interoperability problems between any two WSMO elements. In fact, one core principle behind WSMO is the centrality of mediation as a means to reduce the coupling and deal with the heterogeneity that characterizes the Web.

Together with the ontology, research around WSMO has also produced a family of languages, the Web Service Modeling Language (WSML) [19, 28]. The remainder of this

section covers each of the WSMO elements in more detail. However, for the sake of clarity and simplicity WSML itself is not introduced, instead the Meta Object Facility (MOF) [29], which is the meta-meta model language that has been used for representing the elements of the WSMO ontology is used.

Ontologies, see [Listing 22.2](#), can be defined in a modular way by importing others. When importing ontologies in realistic scenarios, some steps for aligning, merging, and transforming imported ontologies in order to resolve ontology mismatches are required. For this reason, ontology mediators – *OO-Mediators* in particular – are used. The other elements are as normally found within ontology definition languages. *Concepts* constitute the basic elements of the agreed terminology for some problem domain. *Relations* are used in order to model dependencies between several concepts (respectively instances of these concepts); *Functions* are special relations, with a unary range and an n-ary domain (parameters inherited from relation), where the range value is functionally dependent on the domain values, and instances are either defined explicitly or by a link to an instance store, that is, an external storage of instances and their values.

Listing 22.2: The Ontology element in Web Service Modeling Ontology (WSMO)

Class Ontology

```
hasNonFunctionalProperties type nonFunctionalProperties
importsOntology type Ontology
usesMediator type OOMediator
hasConcept type Concept
hasRelation type Relation
hasFunction type Function
hasInstance type Instance
hasAxiom type Axiom
```

Web Services, see [Listing 22.3](#), are online components that provide functionality. Web Services can make use of an extra type of mediator – *WW-Mediator* – to deal with protocol- and process-related mismatches between Web Services. The two core WSMO notions for semantically describing Web Services are *capability* and *service interfaces*.

Listing 22.3: The Web Service element in WSMO

Class WebService

```
hasNonFunctionalProperties type NonFunctionalProperties
importsOntology type Ontology
usesMediator type {OOMediator, WWMediator}
hasCapability type Capability multiplicity = single-valued
hasInterface type Interface
```

A capability defines the functionality offered by a service by means of the following four main items:

- *Preconditions*: A set of logical expressions that specify constraints over the inputs of a service. The focus here is on the accessible data within the available reasoning system.
- *Assumptions*: Within service usage scenarios, one often needs to make statements about the world outside of the platform on which a service is executed. Assumptions provide the means for doing so. Although of course it is not always feasible to check the status value for the statements, the statements are still of value in terms of a formal description of a potentially important constraint.
- *Post-conditions*: A set of logical expressions that specify constraints over the outputs of a service. Like for preconditions, the focus here is on the accessible data within the available reasoning system.
- *Effects*: Statements that relate to the state of the world after the service has been executed. As with assumptions, it may not always be feasible to check the absolute truth values of the statements but still they serve a useful formal documentation role and can facilitate verification and monitoring.

A *service interface* defines how the functionality of a service can be achieved by means of a *choreography* and an *orchestration*. The choreography describes the behavior of a service from the client's point of view. WSMO provides a state-based mechanism for describing choreographies based on Abstract-State Machines (ASMs) [30], whereby the choreography consists of a *state signature* (concepts and variables manipulated) and a set of *transition rules* that manipulate the data.

An *orchestration* captures the control and data-flow within a complex Web Service by interacting with other Web Services. Orchestrations are commonly used to: (a) ensure behavioral congruence, that is, that the orchestration of a service matches its declared choreography, (b) facilitate the reuse of service combinations, and (c) enable client constraints to be checked. The definition of orchestrations in WSMO is still subject of research and is envisioned to be also based on ASMs. Additionally, research has been carried out for providing transformations from more common workflow representation languages such as UML activity diagrams [31].

Goals, see ▶ Listing 22.4, are derived from the notion of task prevalent in previous work including *Knowledge Acquisition and Documentation Structuring* (KADS) [26], UPML [32], and Generic Tasks [33]. Goals are used to represent the viewpoint of a service requester or client. Goals also reflect the structure of a Web Service capturing aspects related to user desires with respect to the requested functionality and behavior. Thus, the requested capability in the definition of a Goal represents the functionality of the services the user would like to have, and the requested interface represents the interface of the service the user would like to have and interact with. Goals therefore represent the starting point for service discovery in WSMO-based frameworks.

Listing 22.4: The Goal element in WSMO**Class** Goal

```

hasNonFunctionalProperties type NonFunctionalProperties
importsOntology type Ontology
usesMediator type {OOMediator, GGMediator}
requestsCapability type Capability multiplicity = single-valued
requestsInterface type Interface

```

Mediators in WSMO handle heterogeneities, which can occur when two software components are put together. Mediators, see ▶ Listing 22.5, are defined on the basis of a number of *sources*, a *target* and a *mediation service* that is in charge of performing the actual mediation. WSMO defines different types of mediators for connecting the distinct WSMO elements: *OO-Mediators* connect and mediate between heterogeneous ontologies, *GG-Mediators* connect Goals, *WG-Mediators* link Web Services to Goals, and *WW-Mediators* connect interoperating Web Services resolving mismatches between them. The mediation service may be specified as a service, a WW-Mediator, or as a Goal.

Listing 22.5: The Mediator element in WSMO**Class** Mediator

```

hasNonFunctionalProperties type NonFunctionalProperties
importsOntology type Ontology
hasSource type {Ontology, Goal, WebService, Mediator}
hasTarget type {Ontology, Goal, WebService, Mediator}
hasMediationService type {Goal, WebService, WWMediator}

```

Following from the extensive use of metadata within the Web, every WSMO element includes a nonfunctional properties attribute that extends the Dublin Core (DC) Metadata Set. Nonfunctional properties include basic information such as the author and creation date and service-specific properties related to the quality of the described service.

22.2.1.2 Bottom-Up Approaches

WSDL-S and SAWSDL

WSDL-S was proposed as a member submission to the W3C in November 2005 between the LSDIS Laboratory at University of Georgia (the majority of the group has since moved to the Kno.e.sis Center, Wright State University, <http://knoesis.org>) and IBM [22]. WSDL-S is a lightweight approach to associating semantic annotations with Web Services

developed in the context of the Managing End-to-End Operations-Semantics (METEOR-S) project that will be presented in more detail in [Sect. 22.2.2.4](#). The key innovation of WSDL-S lies in the use of extensibility in elements and attributes supported by WSDL specification [9]. Using the extensibility of WSDL, semantic annotations in the form of URI references to external models (which can be ontologies, and were broadly termed conceptual models) can be added to the interface, operation, and message constructs. WSDL-S is independent from the language used for defining the semantic models and explicitly contemplates the possibility of using WSML, OWL, and UML as potential candidates [22].

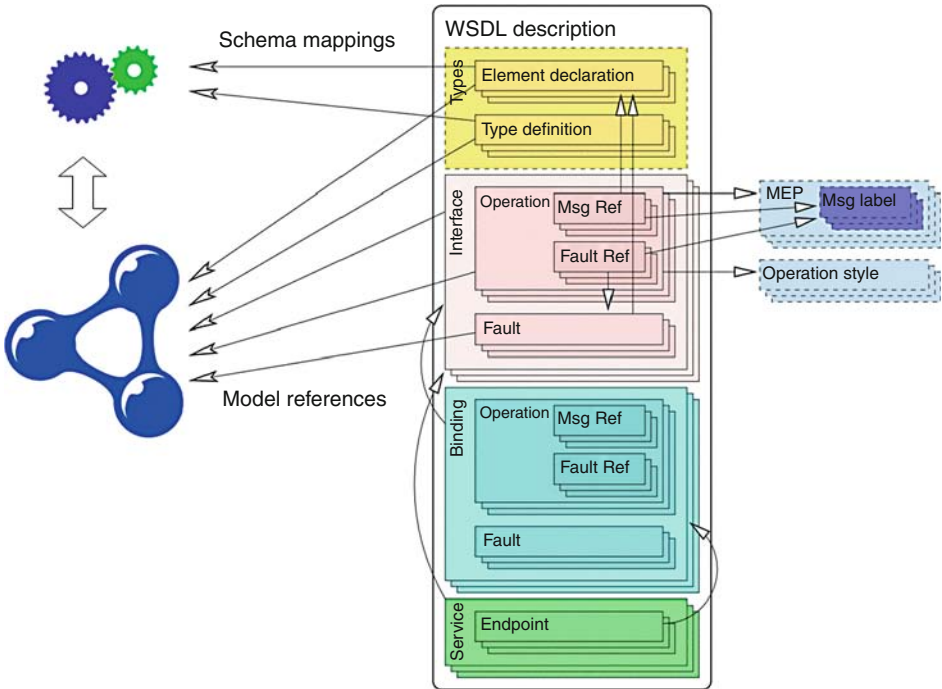
WSDL-S provides a set of extension attributes and elements for associating the semantic annotations. The extension attribute *modelReference* allows one to specify associations between a WSDL entity and a concept in a semantic model. This extension can be used for annotating XML Schema complex types and elements, WSDL operations, and the extension elements *precondition* and *effect*. WSDL-S defines two new children elements for the WSDL operation element, namely *precondition* and *effect*. These elements facilitate the definition of the conditions that must hold before executing an operation and the effects the execution would have. This information is typically used for discovering suitable Web Services.

The *schemaMapping* extension attribute can be used for specifying mechanisms for handling structural differences between XML Schema elements and complex types and their corresponding semantic model concepts. These annotations can then be used for what is referred to as the *lifting* and *lowering* of execution data (i.e., transforming syntactic data into their semantic counterpart and vice versa). The concept of using an intermediate model with lifting and lowering transformation is used as the standard mechanism for mediation by WSDL-S [34].

Finally, WSDL-S includes the *category* extension attribute on the interface element in order to define categorization information for publishing Web Services in registries as defined by the Universal Description, Discovery, and Integration (UDDI) specification [35] for example.

In April 2006, WSDL-S was adopted as the main input for a W3C working group whose task was to create the first W3C recommendation for enabling the semantic annotation of Web Service descriptions. The working group produced the Semantic Annotations for WSDL (SAWSDL) and XML Schema specification [23], which was adopted as a W3C Recommendation in August 2007. SAWSDL is a restricted and homogenized version of WSDL-S including a few changes trying to give a greater level of genericity to the annotations and disregarding those issues for which there existed no agreement among the community at the time the specification was created.

There are essentially three main differences between SAWSDL and WSDL-S. The first one is the fact that *precondition* and *effect* are not directly contemplated since there was no agreement on how to model them within the Semantic Web and Semantic Web Services community. It is worth noting however that SAWSDL does not preclude including these types of annotations as illustrated in the usage guide generated by the SAWSDL working group [36]. Secondly, the *category* annotation is replaced in SAWSDL by the more general *modelReference* extension attribute, which can be used to annotate XML Schema



■ Fig. 22.3

Semantic Annotations for WSDL (SAWSDL) annotations (Figure adapted from [103])

complex-type definitions, simple-type definitions, element declarations, and attribute declarations as well as WSDL interfaces, operations, and faults. Finally, WSDL-S' *schemaMapping* annotation was decomposed into two different extension attributes, namely *liftingSchemaMapping* and *loweringSchemaMapping*, to specifically identify the type of transformation performed. ▶ Figure 22.3 shows a high-level architecture of a SAWSDL annotated Web Service, and ▶ Listing 22.6 gives an example snippet.

Listing 22.6: A SAWSDL snippet describing (parts of) a service supporting the Purchase Order as defined in RosettaNet (URIs have been shortened for reasons of space)

```
<wsdl:description targetNamespace="http://www.w3.org/.../order#"
  xmlns:wsdl="http://www.w3.org/ns/wsdl"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:sawsdl="http://www.w3.org/ns/sawsdl">
  <wsdl:types>
    <xs:element name="processPurchaseOrderResponse" type="xs:string
      sawsdl:modelReference="http://www.w3.org/.../
      rosetta#PurchaseOrderResponse"
```

```

sawSDL:liftingSchemaMapping="http://www.w3.org/.../
POResponse2Ont.xslt"
sawSDL:loweringSchemaMapping="http://www.w3.org/.../
Ont2Response.xslt">
  </xs:element>
</wsdl:types>
<interface name="PurchaseOrder"
  sawSDL:modelReference="http://example.org/.../products/
electronics />
  <operation name="order" pattern=wsdl:in-out
    sawSDL:modelReference="http://www.w3.org/.../
rosetta#RequestPurchaseOrder">
    <input messageLabel = "processPurchaseOrderRequest"
      element="tns:processPurchaseOrderRequest" />
    <output messageLabel = "processPurchaseOrderResponse"
      element="processPurchaseOrderResponse" />
  </operation>
  <operation name="cancel" pattern=wsdl:in-out
    sawSDL:modelReference="http://www.w3.org/.../rosetta#Can-
celOrder" >
    <input messageLabel = "processCancelRequest"
      element="tns:processCancelRequest" />
    <output messageLabel = "processCancelResponse"
      element="processCancelResponse" />
  </operation>
</interface>
</wsdl:description>

```

SAWSDL therefore constitutes a lightweight and incremental approach (compared to OWL-S and WSMO) to annotating WSDL services. In its inception, however, much care was devoted to ensuring its extensibility and on remaining agnostic with respect to the ontologies used, and the languages in which these conceptual models as well as the transformations are defined. As a consequence, SAWSDL has so far been used to link to a variety of ontologies defined in different languages such as RDF(S) and WSML, as well as to point to diverse transformation languages among which XSLT and XSPARQL [37] are perhaps the most commonly applied.

WSMO-Lite

As described in the [Section “WSDL-S and SAWSDL”](#), SAWSDL provides simple hooks for pointing to semantic descriptions from WSDL and XML elements. In particular, it supports three kinds of annotations, namely *modelReference*, *liftingSchemaMapping*, and *loweringSchemaMapping*, which can point to semantic elements described elsewhere on

the Web, or to specifications of data transformations from a syntactic representation to the semantic counterpart and back, respectively. SAWSDL neither advocates a particular representation language for these documents nor provides any specific vocabulary that users should adopt. This characteristic is a means to support extensibility but also forces users to choose their own ontologies for describing services semantically.

WSMO-Lite continues this incremental construction of a stack of technologies for Semantic Web Services by precisely addressing this lack [38]. WSMO-Lite identifies four main types of semantic annotations for services that are a variant of those in [8]:

- *Functional semantics* defines service functionality, that is, the function a service offers to its clients when it is invoked. This information is of particular relevance when finding services and when composing them.
- *Nonfunctional semantics* defines any specific details concerning the implementation or running environment of a service, such as its price or quality of service. Nonfunctional semantics provide additional information about services that can help rank and select the most appropriate one.
- *Behavioral semantics* specifies the protocol (i.e., ordering of operations) that a client needs to follow when invoking a service.
- *Information model* defines the semantics of input, output, and fault messages.

WSMO-Lite provides a minimal RDFS ontology and provides a simple methodology for expressing these four types of semantic annotations for WSDL services using SAWSDL hooks. In particular, to specify the annotations over a concrete WSDL service, WSMO-Lite relies on the SAWSDL *modelReference* attribute for all four semantic annotations, except for information models where *liftingSchemaMapping* and *loweringSchemaMapping* might also be necessary.

WSMO-Lite offers two mechanisms for representing functional semantics, namely simple taxonomies and more expressive preconditions and effects. Functionality taxonomies provide a simple means by which one can define service functionalities through a hierarchy of categories (e.g., *eCl@ss* [39]). This is essentially the typical cataloging approach used in traditional systems such as UDDI, although it is enhanced with reasoning support. In order to distinguish functional classifications from other types of *modelReference* annotations, WSMO-Lite offers the RDFS class *wsl:FunctionalClassificationRoot*, where *wsl* identifies the namespace for WSMO-Lite.

Whenever more expressivity is necessary, WSMO-Lite offers the possibility to enrich functional classification with logical expressions defining conditions that need to hold prior to service execution and capturing changes that the service will carry out on the world. In particular, WSMO-Lite supports defining both by means of the classes *wsl:Condition* and *wsl:Effect*, respectively.

Nonfunctional semantics in WSMO-Lite are represented using external ontologies capturing nonfunctional properties such as security aspects, the quality of service, and price. To do so, WSMO-Lite includes the class *wsl:NonfunctionalParameter* so that

ontologies defining concrete nonfunctional properties for a service, can refer to this class and let the machine know what kind of information it contains.

Behavioral semantics describe how the client should communicate with a service. This kind of description is necessary in order for clients to know, given a particular goal to be achieved, in which order it should invoke the different operations provided by the service. WSMO-Lite, as opposed to its heavyweight counterpart – WSMO – does not include explicit behavioral descriptions. Instead, clients should use the existing functional annotations (classifications, conditions, and effects) over the entire service and the internal operations in order to figure out in which order to invoke them. For instance, planning-based techniques could be applied locally to compute the order for invoking operations. This approach gives total flexibility to both clients and service annotators.

Finally, the information model captures the semantics of the data exchanged between a service and its clients. Indeed, understanding the data is crucial for automated invocation as well as for Web Service compositions where data mediation may be necessary. WSMO-Lite relies on external ontologies for capturing information models. Information models are distinguished from other models such as functional classifications by using *wsl:Ontology*. Additionally, because Web Services generally work with application-specific XML data, WSMO-Lite advocates the use of *liftingSchemaMapping* and *loweringSchemaMapping* for pointing to transformation specifications alongside the model references on the appropriate XML Schema components.

MicroWSMO

Websites are increasingly offering services and data through Web APIs and RESTful services [40], that is, services adopting REST principles [5]. These services are combined by Web developers into what is usually referred to as mash-ups, which obtain data from various sources and process the data in order to generate all sorts of enriched data visualizations like annotated maps, for supporting the integration of social websites, etc.

This type of service is generally described using plain, unstructured HTML, except for a few that use the XML-based format Web Application Description Language (WADL) [40]. As a consequence, despite their popularity, the development of Web applications that integrate disparate services in this manner suffers from a number of limitations similar to those previously outlined for Web Services with the increased complexity that most often no machine-processable description is available. Discovering services, handling heterogeneous data, and creating service compositions are largely manual and tedious tasks that end up in the development of custom-tailored solutions that use these services.

In the light of the popularity and limitations of this technology, research on Semantic Web Services has recently focused on trying to support further automation within the life cycle of applications based on Web APIs and RESTful services. MicroWSMO is a microformat supporting the semantic annotation of RESTful services and Web APIs in order to better support their discovery, composition, and invocation. Microformats offer means for annotating human-oriented Web pages in order to make key information machine-processable [41].

MicroWSMO builds upon hRESTS (HTML for RESTful services) [42]. hRESTS enables the creation of machine-processable Web API descriptions based on available HTML documentation. hRESTS provide a number of HTML classes that allow one to structure APIs descriptions by identifying services, operations, methods, inputs, outputs, and addresses. It therefore supports, by simple injections of HTML code within Web pages, the transformation of unstructured HTML-based APIs descriptions into structured service descriptions similar to those provided by WSDL. Listing 22.7 shows an example of an hRESTS description.

Listing 22.7: Example of an hRESTS (HTML for RESTful services) annotation

```
<div class="service" id="s1"><h1>happenr API</h1>
  <span class="label">Happenr </span>has two main methods to call
    "getEvents" and . .
  <p>All operations should be directed at http://happenr.3scale.
    net/</p>
  <h2>Example usage</h2>
  <span class="address">
    http://happenr.3scale.ws/.../getEvents.php?user_key=xxx
  </span>
  <p>where the userkey is the key issues with the signup you made.</p>
  <div class="operation" id="op1">
    <h2><span class="label">getEvents </span>Method</h2>
    <span class="input"><h3>username</h3>
      <p>Your username that you received from Happenr ...</p>
      <h3>password</h3>
      <p>Your password that you received from Happenr ...</p>
      <h3>eventId</h3>
      <p>The id of the event.</p>
    </span>
  </div>
</div>
```

With the hRESTS structure in place, HTML service descriptions can be annotated further by including pointers to the semantics of the service, operations, and data manipulated. To this end, MicroWSMO extends hRESTS with three additional properties, namely *model*, *lifting*, and *lowering* that are borrowed from SAWSDL and have the same semantics explained earlier. Although not strictly necessary, MicroWSMO adopts the WSMO-Lite ontology as the reference ontology for annotating RESTful services semantically. By doing so, both WSDL services and RESTful services annotated with WSMO-Lite and MicroWSMO respectively can be treated homogeneously. See Listing 22.8 for an example of a MicroWSMO annotation.

Listing 22.8: Example of a MicroWSMO annotation

```

<div class="service" id="s1"><h1>happenr API</h1>
  <a rel="model" href="http://example.com/events/getEvents">
    <span class="label">Happenr </span>has two main methods to
    call
    "getEvents" and . .
  </a>
  <p>All operations should be directed at http://happenr.3scale.
  net/</p>
  <h2>Example usage</h2>
  <span class="address">
    http://happenr.3scale.ws/webservices/getEvents.php?
    user_key=xxx
  </span>
  <p>where the userkey is the key issues with the signup you
  made.</p>
<div class="operation" id="op1"><h2>
  <span class="label">getEvents </span>Method</h2>
  <span class="input">
    <h3>
      <a rel="model"
        href="http://example.com/data/onto.
        owl#Username">username</a>
      (<a rel="lowering"
        href="http://example.com/data/event.
        xsparql">lowering</a>)
    </h3>
    <p>Your username that you received from Happenr ...</p>
    <h3>
      <a rel="model"
        href="http://example.com/data/onto.
        owl#Password">password<a>
      (<a rel="lowering"
        href="http://example.com/data/event.
        xsparql">lowering</a>)
    </h3>
    <p>Your password that you received from Happenr in order to query
    this webservice.</p>

```

Like WSMO-Lite, MicroWSMO incorporates four main types of semantic annotations for services: functional semantics, nonfunctional semantics, behavioral semantics, and information model semantics. Functional and nonfunctional semantics are to be provided

through model references on the service. Behavioral semantics are included as model references on the operations. Finally, information model semantics are captured on the input and output messages of operations. The reader is referred to the WSMO-Lite description for further details [38].

SA-REST

SA-REST [43, 44] is an open, flexible, and standards-based approach to adding semantic annotations to RESTful services and Web APIs. SA-REST essentially borrows the idea of grounding service descriptions to semantic metamodels by using model reference type of annotations from SAWSDL. However, this is not all there needs to be since, like MicroWSMO, SA-REST does not start from a machine-processable description of a RESTful service or Web APIs similar to WSDL files. In most of the cases, Web APIs are solely described in human-oriented HTML Web pages, which do not include elements that a machine could directly use for identifying services and their elements.

Consequently, SA-REST uses microformats as a means to embed semantic annotations within the Web pages describing RESTful services. In particular, it supports the use of GRDDL [45] and RDFa [46], which are both W3C Recommendations. The former offers a way for choosing any microformat and specifying a translation of the Web page into machine-processable text. The latter supports embedding RDF within XML, XHTML, and HTML. The SA-REST specification, however, recommends using RDFa since it supports keeping the entire description of the service, may it be the human-readable text or the machine-processable RDF, within one single document.

SA-REST leaves up to the user as to how to embed the RDF triples within the text. They could therefore be spread across the document or clustered together. The triples embedded are such that the subject should be the URL at which the service is invoked. The predicate should be one of *sarest:input*, *sarest:output*, *sarest:operation*, *sarest:lifting*, *sarest:lowering*, and *sarest:fault* whereby *sarest* corresponds to the SA-REST namespace. The triple's object should be either a URI or a URL pointing to a resource, which in the case of *sarest:lifting* and *sarest:lowering* will be a transformation and for the others it will be an element from an ontology. Example annotations for Craigslist search service are illustrated in [Listing 22.9](#).

Listing 22.9: SA-REST annotations for Craigslist search service

```
<html xmlns:sarest="http://lstdis.cs.uga.edu/SAREST#">
...
<p about="http://craigslist.org/search/">
  The logical input of this service is an
  <span property="sarest:input">
    http://lstdis.cs.uga.edu/ont.owl#Location_Query
  </span>
  object. The logical output of this service is a list of
  <span property="sarest:output">
```

```

        http://lstdis.cs.uga.edu/ont.owl#Location
    </span>
    objects. This service should be invoked using an
    <span property="sarest:action">
        HTTP GET
    </span>
    <meta property="sarest:lifting"
content="http://craigslist.org/api/lifting.xsl" />
    <meta property="sarest:lowering"
content="http://craigslist.org/api/lowering.xsl" />
    <meta property="sarest:operation"
content="http://lstdis.cs.uga.edu/ont.owl#Location_Search" />
</p>

```

Pertinent research that builds upon SA-REST includes the faceted search of API documents [47] including a ranking algorithm called ServiUt where annotations are used to build a searchable, comprehensive classification of API documents.

22.2.2 Semantic Web Services Infrastructure

Alongside the conceptual models described in the [Sect. 22.2.1](#), there has been extensive development trying to exploit the conceptual models to automate some of the core tasks for handling Web Services. In the remainder of this section, some of the main systems are covered organized by framework – when there exist several tools and engines belonging to the same framework – and by the conceptual model they build upon. OWL-S-related tools are covered first, then the main WSMO platforms are presented, and finally the work around WSDL-S and SAWSDL is described.

22.2.2.1 OWL-S Tools and Engines

Research focusing on tool development for OWL-S has largely taken place as separate elements or components developed by different research groups in a stand-alone basis rather than as a full-fledged framework covering the entire life cycle of Semantic Web Services. Among the main research and development around OWL-S, there has been effort devoted to implementing matchmaking engines, editors, execution environments, or even (semi) automated composition engines. A number of these tools are described in the remainder of this section.

Annotation

One of the very first OWL-S editors is a plug-in for Protégé [48], which extended the ontology editor toward supporting the definition of Semantic Web Services based on OWL-S. It therefore benefits from the general ontology editing features Protégé provides,

extends, and includes OWL-S-specific panes structured around the OWL-S subontologies capturing the Service Profile, the Service Model, and the Service Grounding. The editor includes additional features for supporting the management of inputs, outputs, preconditions, and effects, as well as graphical support for defining the control-flow of processes based on OWL-S inbuilt constructs.

A second editor that is worth mentioning is ASSAM, which stands for Automated Semantic Service Annotation with Machine learning. ASSAM as opposed to most Semantic Web Services editors is a tool for the semiautomatic annotation of Web Services [49]. In a nutshell, the tool assists the user in annotating Web Services relying on two machine-learning algorithms. The first algorithm aids in annotating datatypes in WSDL files by classifying services, their operations, and the messages they exchange given an initial training set of previously annotated services. The second algorithm helps the user in mapping schemas for different yet related Web Services. The resulting annotations can eventually be exported in OWL-S. Currently ASSAM constitutes one of the most advanced Semantic Web Services annotation tools, together with the research carried out by Sabou [50].

Matching

Work on service matching based on OWL-S has largely been based on the use of subsumption reasoning supported by OWL and related reasoners. Among the first service matchmakers that were developed in the area the DAML-S Matchmaker [51, 52] developed at Carnegie Mellon could be cited, which was used to develop an enhanced UDDI registry. The registry maps OWL-S profiles to UDDI defining specialized T-Models for OWL-S-specific elements that cannot be mapped to UDDI directly such as inputs, outputs, for example. To support advanced matching, each advertisement is preprocessed in order to derive the corresponding UDDI advertisements out of OWL-S profile definitions and also to define the degree of matching with respect to ontology concepts using RACER [53] as the OWL reasoning engine. Through this simple approach, the enhanced UDDI registry is able to support more advanced matching contemplating situations where requests are less specific than advertisements (plug-in match), and where requests are more specific than advertisements (subsume match) thanks to the use of subsumption reasoning. At querying time, the degree of matching is established depending on the number of intervening classes between requests and advertisements.

The OWL-S Web Service Matcher [54] (OWLS-M), is another matchmaking engine. In this case, the engine carries out the service matchmaking in a four-step process. The first two steps consider the input and output types of a service and perform subsumption reasoning between the requests and the known services. The third step takes into account the categorization of the service itself and finally OWLS-M facilitates the use of custom elements addressing individual constraints or requirements (e.g., quality of service) to filter the results.

The OWL-S Service Matchmaker (OWLS-MX) [55] is a hybrid OWL-S matchmaker that exploits logic-based techniques and Information Retrieval syntactic similarity measures for approximate matching of services. In a nutshell, the OWLS-MX takes any OWL-S service as a matchmaking query, and returns an ordered set of relevant services

including the degree of match and syntactic similarity. Matchmaking is carried out by processing every input and output of the services contemplated, doing subsumption checking and similarity comparisons with the inputs and outputs specified in the request.

OWLS-MX provides five different matching filters, three of which are logic-based namely *Exact*, *Plug-in*, and *Subsumes*, and two of which are hybrid, namely *Subsumed-by*, and *Nearest-neighbor*. Combined, they allow users to carry out approximate service matchmaking with varying degrees of match ranging from exact logic-based matching to approximate neighborhood based on syntactic similarity. The user is provided the means for specifying a threshold for syntactic similarity that establishes that any failure in subsumption checking will be tolerated if the syntactic similarity obtained is beyond the threshold.

In addition to the work mentioned so far, some researchers have approached service matchmaking in OWL-S purely from a Description Logics (DL) perspective, perhaps the most notable example being [56]. In a nutshell, the idea is to treat both service advertisements and queries as concepts and then consider the subsumption relation between both. Based on this approach, Li and Horrocks [56] have been able to develop a generic service matchmaking engine for DAML-S using RACER [53] – a general-purpose DL reasoner. Although treating advertisements as concepts is somewhat counterintuitive, the authors showed that there is no noticeable expressivity impact and conversely there is a clear processing advantage. The matchmaking engine developed in this manner is able to provide the typical range of matching degrees namely exact, plug-in, subsume, and fail (called disjoint in this case) and an additional one – intersection – when the advertisement and the query are almost but not completely incompatible.

Orchestration

At the core of OWL-S framework is the OWL-Virtual Machine (former DAML-S VM) [52, 57], a general-purpose Web Service client, which relies on the OWL-S process model and grounding to support interaction between OWL-S Web Services. At the core of the OWL-S Virtual Machine lies an OWL-S Processor, which is supported by an OWL Inference Engine that combines Jena [58] and the Jess engine [59] for including rules support. The OWL-S Processor is in charge of driving the interaction with services and therefore implements the operational semantics of the OWL-S Process Model by means of rules and relying on a Web Service invocation module for interacting with remote services.

Composition

The OWL-S Virtual Machine was also utilized to explore the possibility for automatically composing and executing OWL-S processes. The research, which was carried out by extending an existing planning engine, although preliminary, highlighted to an extent the potential and also the outstanding challenges that had to be tackled in this area. In the light of these issues, and the inherent computational complexity of planning algorithms, Sirin et al. developed a semiautomated service composition tool [60]. Essentially the tool allows humans to search and filter services based on the DAML-S Matchmaker techniques allowing them to choose the most appropriate services to compose new processes. Thus, by delegating the core decisions to humans the tool reduces the inherent

computational complexity of process composition while it leverages semantic descriptions to better support the creation of processes. The resulting processes can subsequently be exported as OWL-S Composite Processes.

In [61], Traverso and Pistore present one of the main planning solutions devised for supporting the automated composition of services described in OWL-S. Their approach is able to deal with nondeterminism, for example, the fact that a service invocation may return a result or an error, partial observability accounting for the fact that one can only observe communications with services and not their internal variables, as well as complex goals. The result of the composition is an executable Business Process Execution Language (BPEL) process. In a nutshell, the solution they propose is based on the translation of OWL-S process models into state transition systems and subsequently applying knowledge-level planning algorithms that aim to fulfill the requested composition goal. In addition to the actual formalization and application of AI planning algorithms to OWL-S, perhaps the most notable outcome of their research is the fact that they illustrate how the use of semantic annotations can help improve the performance of Web Service composition.

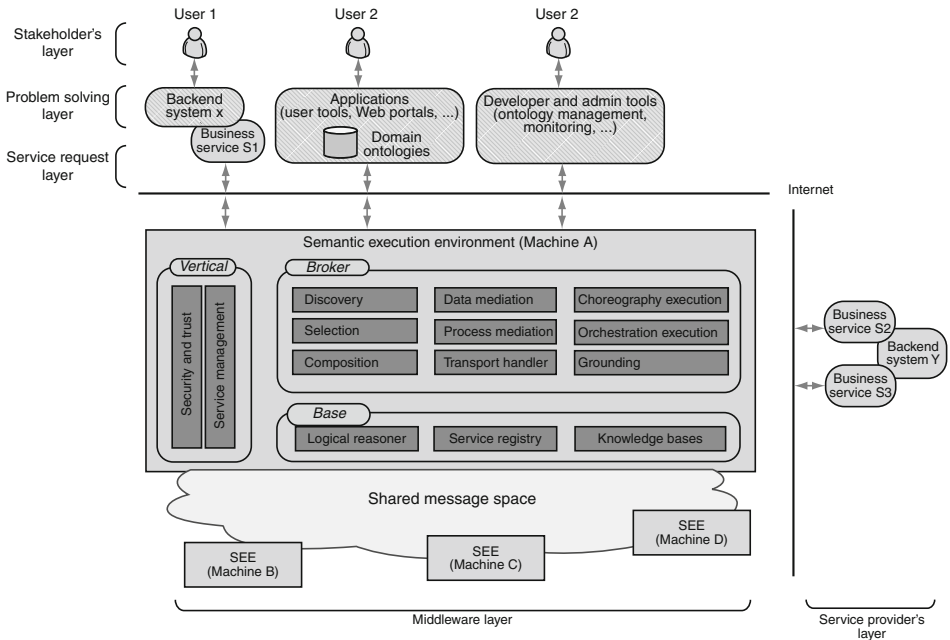
22.2.2.2 WSMX

Research and development on WSMO has dedicated substantial efforts to developing engines and frameworks able to interpret WSMO descriptions and use them for achieving a greater level of automation during the life cycle of service-oriented applications. The Web Service Execution Environment (WSMX) [15] is, together with IRS-III described in [▶ Sect. 22.2.2.3](#), one of the reference implementations of WSMO and also of Semantic Execution Environments (SEEs) being standardized within OASIS [13].

WSMX is a component-based execution environment for WSMO that aims to support the discovery, composition, mediation, selection, and invocation of Web Services based on a set of user's requirements. The main effort associated with WSMX has been on defining the mandatory functional services that need to be used within each of these particular tasks as well as their interfaces.

WSMX has been defined as a layered architecture depicted in [▶ Fig. 22.4](#), whereby at the bottom layer reside formal languages for representing semantic information (WSML in this case) and the machinery for reasoning and storing this knowledge. The middle layer provides brokering facilities necessary for manipulating Semantic Web Services descriptions. This layer therefore comprises components for discovery, selection, data and process mediation, choreography, orchestration, grounding, and transport handling. The top layer is the interface to external applications and users and is therefore the layer where domain ontologies, applications, and developer tools reside. Finally, a vertical layer across all three layers is in charge of managing the execution as well as security and authentication aspects.

The degree of development and refinement for each of these components is quite heterogeneous since the focus was put on the definition of a generic architecture and the different execution scenarios, such as goal achievement and service selection. In the remainder of this section, the main aspects behind some of the core components are



■ Fig. 22.4

Web Service Execution Environment (WSMX) architecture (Taken from the OASIS Semantic Execution Environment Technical Committee)

introduced. The reader is referred to [15] for more concrete details on each of these and the underlying approaches adopted.

Annotation

As previously introduced, WSMX is based on WSMO for which a specific family of representation languages, WSML, was defined. SWS descriptions processed by WSMX, as well as the respective domain ontologies are expressed in WSML terms. Consequently, substantial efforts have been devoted to providing full-fledged development environments able to support users in defining WSMO entities including Ontologies.

The two main frameworks developed, were WSMO Studio [62] and the Web Service Modeling Toolkit (WSMT) [63]. The features provided by both eclipse-based frameworks are quite similar. They include support for modeling all four top-level elements of WSMO as well as visualization support for ontologies. Also, they both integrate WSML reasoners so that developers can validate ontologies or evaluate queries over knowledge bases. Finally, they include support for interacting with ranking and selection engines and there is also support for creating mapping definitions in order to define Mediators.

Discovery and Matching

In ▶ Sect. 22.2, a somewhat singular definition for *discovery* was introduced, which contrasts with the functionality-oriented view often adopted by researchers in the field.

This definition is based on an understanding of discovery anchored on the distinction between services as business entities and Web Services as computational entities that allow clients to interact with providers in order to benefit from (business) services. This distinction, which has been highlighted in several occasions by other researchers [11, 64], was investigated in WSMX.

Driven by this understanding of discovery and by the core notions of WSMO, WSMX proposes a conceptual model for service discovery (as opposed to Web Service discovery) that approaches this endeavor as a *heuristic classification* task [65]. Heuristic classification is a problem-solving method that was identified as being widely applied for carrying out tasks such as *classification* and *diagnosis*. It essentially separates the task into three main steps: *abstraction*, *matching*, and *refinement*. Abstraction is in charge of abstracting the essential features of a given case or situation. The second step is in charge of *matching* the abstracted features with prototypical cases. Finally, refinement proceeds toward a final solution by trying to explain all the particularities of the given case given all the potential matches identified.

WSMX views discovery as a process starting from some desires out of which a set of Goals are *abstracted*. Once the Goals have been identified, they are *matched* against known Web Services (what they call Web Service discovery) and finally each of the Web Services matched is *checked* to determine whether they can fulfill the original desires (what they call service discovery).

Most of the implementation work on discovery in WSMX has focused on one of the steps of the conceptual model described above, namely Web Service discovery, that is, the matching of abstract Goals to Web Services, leaving the abstraction and refinement processes somewhat aside. Among the work so far it is worth mentioning the theoretical work exposed in [15] tackling the problem with different expressivity ranging from simple keywords up to advanced Web Service discovery based on rich Web Services descriptions and transaction logic.

It is also worth mentioning the research carried out by Stollberg et al. [66], which, as opposed to most of the work around SWS matching, focused on achieving performance and scalability at runtime rather than on improving accuracy. This work is based upon an extension of the WSMO model that distinguishes between the notion of *Goal Templates* and that of *Goal instances*, which is inspired by the treatment of Goals as meta-classes in IRS-III, which is described in more detail in [Sect. 22.2.2.3](#). Goal Templates are used for generating a Semantic Discovery Caching graph that organizes Goal Templates in a hierarchy based on their similarity from a functional perspective. Then, at runtime this caching graph is exploited in order to minimize the discovery time even for large repositories of Web Services.

Finally, Glue [67] is another implementation of Web Service discovery for WSMX. Glue makes use of a number of extensions to WSMO to support its objectives. First, Glue's model defines the class of Goals and the class of Web Services in a similar fashion to that of IRS-III. Second, the GG-Mediators (see [Sect. "WSMO"](#)) are used for automatically generating a set of goals semantically equivalent to the one expressed by the requester but expressed with a different form or using different ontologies. Third, WG-Mediators are

used for evaluating the matching between Goals and Web Services and OO-Mediators are explicitly incorporated in the discovery process in order to deal with semantic mismatches. Finally, the discovery process is extended in order to include both the matchmaking and the discovery of Mediators that could resolve heterogeneities.

Ranking and Selection

As part of WSMX, a ranking and selection engine has been developed. The engine is able to take into account nonfunctional annotations associated with Web Services in order to, once they have been selected as matching a Goal, rank them accordingly. The engine contemplates general annotations such as the creator of the annotation and the publisher as well as nonfunctional properties of the service such as its availability or its price.

Together with the engine, there are a set of ontologies enabling the capture of nonfunctional properties. Among the ontologies provided, the system includes conceptualizations covering location, time, price, availability, trust, and quality of service. On the basis of these ontologies, one can define nonfunctional properties for services using logical expressions and at selection time the engine can check the values for the matching services in order to rank them. The current implementation takes as input the user preferences expressed as logical expressions as well as the matching services and makes use of a multi-criteria algorithm to rank the services accordingly.

Orchestration

Research on orchestration in WSMO and consequently on its interpretation has essentially focused on two orthogonal problems. On the one hand, effort has been devoted to minimizing the complexity for modeling workflows while supporting the mapping into formal representations that can support reasoning about process behavior. In [31], Norton et al. propose a three-level approach to orchestration definition, which provides a layer based on UML Activity Diagrams for supporting process modeling using state-of-the-art editors that are well known by software developers. The process modeling constructs are defined by the Cashew workflow language, which provides support for block-oriented compositions largely inspired by OWL-S. Processes represented in Cashew can be interpreted directly or they can be transformed into Abstract-State Machines for this purpose.

On the other hand, other research has focused on providing a solution with existing workflow standards while supporting the application of semantics to support process adaptability among others. The work carried out in this respect is based on the use of BPEL for defining workflows that can directly refer to Web services or that can support the inclusion of Goals so that at runtime and given the concrete situation, the most promising service can be used [68].

Mediation

Previously, the fact that one of the core features of WSMO is the central role played by mediation was introduced. Mediation is brought into WSMO models by means of four main constructs, namely OO-Mediators, GG-Mediators, WG-Mediators, and WW-Mediators.

Data mediation in WSMX takes place at two main stages [15]. The first one concerns data representation and is supported by the inclusion of lifting and lowering mechanisms. On the basis of these definitions, at runtime WSMX is able to transform data from their internal semantic representation in WSML to and from the XML-base syntactic representation used by Web Services. Doing so, therefore, avoids issues between heterogeneous XML representations by using an intermediate transformation into WSML. The second stage takes care of semantic heterogeneity by means of an abstract mapping language supporting the declarative specification of mappings between two different ontologies [102]. The language is supported by the WSMT modeling environment through a semiautomated editor. The resulting mappings can be interpreted at runtime by an engine in order to perform the appropriate transformations.

Work on process mediation aims at mediating between two communicating Web Services so that differences in the kinds of messages and their order do not prevent them from communicating effectively. Research on process mediation in WSMX has produced a categorization of the different mismatches that can be found and a process mediator that can resolve those that are deemed solvable. The categorization includes solvable mismatches such as stopping unexpected messages, splitting a message, combining messages, or inverting messages to name a few. The unsolvable mismatches occur when some piece of information expected by one of the actors cannot be sent by the other or when both actors expect data from the other. In a nutshell, the prototype developed solves these process mediation problems by keeping track of both Web Service choreographies, checking the messages expected and the data available in both sides so as to forward the necessary information to the other Web Service when appropriate. Indeed the process mediator relies on existing data mediation information in order to solve mismatches like the splitting or merging of messages.

22.2.2.3 IRS-III

The Internet Reasoning Service (IRS) project carried out at the Knowledge Media Institute of The Open University has the overall aim of supporting the automated or semiautomated construction of semantically enhanced systems over the Internet. IRS-I supported the creation of knowledge-intensive systems structured according to the Unified Problem-solving Method Development Language (UPML) [27]; IRS-II [69] integrated the UPML framework with Web Service technologies. IRS-III extended this framework incorporating the WSMO conceptual model, and providing additionally a set of tools to support the SWS developer at design-time in creating, editing, and managing a library of semantic descriptions as well as publishing and invoking Semantic Web Services [70].

IRS-III is a broker-based platform that mediates between clients and service providers allowing each of them to adopt the most convenient means for representing their perspective, while supporting an effective interaction. To this end, IRS-III is entirely underpinned by ontological representations facilitating knowledge sharing by machines and humans.

IRS-III uses its own ontology representation language, OCML [71]. The OCML language combines a frame system with a tightly integrated forward- and backward-chaining rule system and includes constructs for defining classes, instances, relations, functions, procedures, and rules. Additionally, procedures and functions can be attached to Lisp code. This feature allows ontologies related to service descriptions to be attached to the IRS service invocation mechanism, thus enabling inferred values to reflect the state of a deployed service (e.g., to retrieve a current exchange rate). In order to ensure its interoperability, OCML contains import/export facilities to RDF(S) and WSML and import facilities for OWL [70].

IRS-III is based on a service ontology, which influenced as well as it was informed by WSMO presented earlier in this chapter, and is therefore largely aligned with it. Hence, the IRS-III service ontology contemplates *Goals*, *Web Services*, and *Mediators* as the main concepts, and Web Services have *Choreographies* and *Orchestrations* specifying how to communicate with them and how they combine other services to achieve their functionality, respectively. Still, there exist fundamental differences between both ontologies that are worth outlining.

First, the IRS-III service ontology uses meta-classes for the top-level SWS concepts (Goal, Mediator, and Web Service). As a consequence, IRS-III components can reason over the top-level concepts within the service ontology as first-class entities, which is something not directly possible in WSML-based representations of WSMO where special keywords as opposed to ontological entities are used. On the basis of these concepts, IRS-III allows users to define the required Goals, Mediators, and Web Services as subclasses of the corresponding WSMO concepts rather than as instances, supporting the creation of reusable service descriptions and taxonomic structures that can be exploited for reasoning and indexing purposes.

Additionally and based on the previous distinction, the invocation of Semantic Web Services are kept as instances. When IRS-III receives a client request, instances of relevant Goals, Mediators, and Web Services are created to capture the current invocation context. Doing so paves the way for carrying out thorough analyses and monitoring of the invocations, as well as allowing the implementation of caching mechanisms.

Finally, in the interest of simplifying the definition of Goals and Web Services, the service ontology incorporates explicit input and output role declarations. The declared input and output types are imported from domain ontologies. This feature enables developers to view Goals and Web Services as “one-shot” invocable entities, thus minimizing the need to consider complex interactions when appropriate.

One distinctive feature of IRS is the support it provides for the seamless publishing of services. Indeed, users quite often will have an existing system functionality, which they would like to be made available as a service, but have no knowledge of the tools and processes involved in turning a stand-alone program into a Web Service. To cater for this IRS-III provides support for “one-click” publishing of stand-alone code (currently Java and Lisp), as well as Web Services from a WSDL description or from the URI (i.e., a HTTP GET request) of a Web application.

Significant effort has been devoted toward supporting the interoperability between IRS-III and other Semantic Web Services frameworks. In particular, IRS-III has an OWL-S

import mechanism [72] and is interoperable with WSMO implementations (e.g., WSMO Studio [62] and WSMX [15]) through a common API [13] in line with the standardization within OASIS of the Semantic Execution Environment.

From a technical perspective, as can be seen in [Fig. 22.5](#), the *IRS-III Server* builds upon an *HTTP Server* written in Lisp, which has been extended with a SOAP handler. At the heart of the server is the *SWS Library*, where the semantic descriptions associated with Web Services are stored using OCML as the representation language. The library is structured into domain ontologies and knowledge models for Goals, Web Services, and Mediators as described previously.

Annotation

IRS-III provides its own development environment called the IRS Browser, which supports users in defining WSMO entities and uploading them into the IRS directly. The browser additionally supports the direct invocation of Web Services or Goals. Furthermore, based on the work carried out on interoperability between Semantic Execution Environments, IRS-III implements the APIs defined, which therefore ensures its interoperability with existing environments like WSMT and WSMO Studio introduced earlier. It is therefore possible to use both editing environments to retrieve and store any WSMO entity to and from the IRS, and to use it for invoking Goals directly.

Ranking and Selection

IRS-III provides capability-based invocation allowing users to request a Goal to be achieved by instantiating a Goal definition with concrete values (e.g., Book transportation

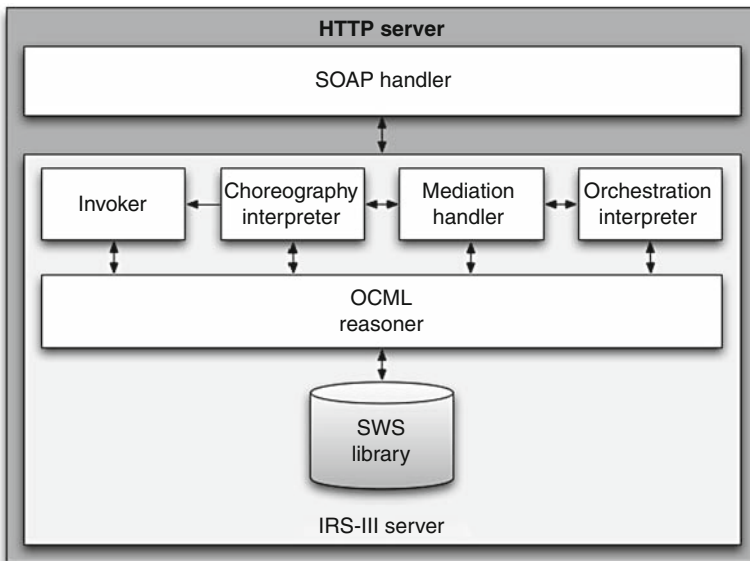


Fig. 22.5

Architecture of IRS-III

from *London* to *Manchester* for *next Tuesday morning*). Upon reception of a Goal request, IRS-III explores its library of known services in order to identify all the Web Services that are suitable, and then select the one that is considered the best.

Research on IRS-III has therefore not focused on Web Services discovery by crawling the Web or existing UDDI registries. Instead, research has been concentrated on supporting an effective ranking and selection of appropriate services based on a variety of criteria. To this end, IRS-III leverages the WG-Mediator definitions that link Goals to Web Services to identify suitable and directly invocable services and the existing hierarchies of Goals to provide more refined or relaxed solutions should a perfectly fitting service not be available or known.

This simple selection mechanism has been extended toward supporting more refined mechanisms based on arbitrary criteria over nonfunctional parameters. This work is based on the application of a domain-independent library of Heuristic Classification [65] Problem-Solving Methods implemented in OCML [73]. As mentioned earlier, Heuristic Classification is a method for classifying entities that *relates data to a pre-enumerated set of solutions by abstraction, heuristic association, and refinement* [65].

The library provides a task ontology that defines the kinds of knowledge necessary to classify things according to a *taxonomy of classes* given a set of *observed features*. Based on raw information, classification criteria as well as additional domain-specific knowledge including heuristics, the library is able to automatically classify the situation at hand with respect to the target taxonomy.

This library has been used for developing a trust-aware service selection mechanism [74] that takes into account *user preferences*, *trust requirements*, and *trust guarantees* promised by service providers in order to choose those services that best meet the trust requirements out of those that are functionality adequate. To this end, the solution implemented provides two main classification methods: a *single solution classification* method based on a hill-climbing algorithm; and an *optimal classification* method based on exhaustive search.

Although this selection mechanism is solely based on trust information, work has been carried out more recently in order to benefit from the genericity of the approach [75]. In particular, the aforementioned techniques have been extended toward supporting a context-aware selection of services that can take into account observed/monitored features of services and users across arbitrary and composable dimensions (e.g., time, trust, location), so as to refine the set of functionally suitable services into the most contextually relevant subset.

Choreography

IRS-III offers capability-based invocation, that is, the ability to invoke services based on the client request expressed as a Goal. IRS-III then acts as a broker: finding, composing, and invoking appropriate Web Services in order to fulfill the request. In this way, IRS maintains a clean separation between users and services but still supports the invocation of services. Choreography addresses the problem of communication between a client and a Web Service. Since the IRS-III acts as a broker, the choreography work in IRS-III is focused on the communication between the IRS-III and the relevant deployed Web Services.

Goal achievement consists of a number of discrete steps, in which, at any given point of the execution, the next action performed will depend upon the current state. IRS-III, adopts the Abstract-State Machine (ASMs) formalism [30] to represent the IRS-III interaction with a client (choreography) or providing Web Services (orchestration).

A choreography is described in IRS-III by the declaration of a *grounding* and a set of *guarded transitions*. The grounding specifies the conceptual representation of the operations involved in the invocation of a Web Service and their mapping to the implementation level. More specifically, the grounding definitions include the operation name and bindings for input and output data. Guarded transitions can be seen as ASM transition rules. These represent the interaction between IRS-III and the Web Service and are applied when executing the choreography. This model is executed at a semantic level when IRS-III receives a request to achieve a Goal.

The IRS-III service ontology defines a set of choreography-specific primitives, which can be used in transition rules. The primitives provided include *init-choreography*, *send-message*, *receive-message*, *send-suspend*, *receive-suspend*, *receive-error*, and *end-choreography*. These primitives provide an easy to use interface to control a conversation between IRS-III and a Web Service. Developers are also able to include any relation defined within the imported ontologies in guarded transition specifications.

The IRS-III uses the OCML forward-chaining-rule engine to execute a choreography. This means that rules belonging to a choreography are fired according to the state taking into account inferences at the ontological level. During communication with a Web Service, IRS-III provides lifting and lowering mechanisms mapping the ontological level descriptions to the XML-based representations used by the specific Web Service invoked. The actual lifting and lowering definitions are based on Lisp macros mapping XPath expressions to ontological elements.

Orchestration

Research and development on orchestration in the IRS has focused on supporting the creation and visualization of orchestrations by developers [31] and on executing them. In IRS-III, the orchestration is used to describe the model of a composed Web Service. At the semantic level, the orchestration is represented by a workflow model expressed in OCML. The main characteristics of the IRS-III orchestration model are:

- *Goal-centric composition*: The basic unit within compositions is a Goal. The orchestration model thus provides control and data-flow constructs over sets of Goals.
- *Invocation is one-shot*: It is assumed that when a Goal is invoked the result is returned and there is no direct interaction between any of the underlying Web Services involved. IRS-III acts as a broker and therefore a dialog between two Web Services becomes a pair of IRS-III to Web Service choreographies.
- *Orchestration is layered*: Within the IRS-III, there are a number of layers each of which supports a specific set of activities [31].

Based on the aforementioned principles, IRS-III provides a set of control-flow primitives, namely *orch-sequence*, *orch-if*, *orch-repeat*, *orch-get-goal-value*, and *orch-return*, which

have been implemented and mapped into ASMs. At runtime, IRS-III provides the capability to interpret orchestrations defined using these constructs by resolving Goals to concrete Web Services and dealing with data-flow mismatches on the basis of Mediators that are described in more detail next.

Mediation

IRS-III provides support for data mediation by supporting the modeling of specialized mediators, which provide a mediation service or declarative mappings for solving different types of conceptual mismatches. The mediation handler interprets each type of mediator accordingly during selection, invocation, and orchestration. The mediator models are created at design-time and used at runtime.

In IRS-III, application developers can associate a relation mapping to OO-Mediators (i.e., Mediators between ontologies) in order to map between instances of different ontologies using declarative expressions in OCML. Additionally, the remaining kinds of mediators (WW-Mediator, WG-Mediator, and GG-Mediator) can be associated with a mediation function encapsulated as a standard Semantic Web Service in order to perform transformations between inputs and outputs. These mediators can provide mediated data-flow between a Goal and a Web Service, and between Web Services or Goals by relying in turn on specific OO-Mediators that declaratively specify the mappings.

Mappings specifications are expressed in OCML based on three main primitives [70, 71]:

- *Maps-to*: A relation created internally for every mapped instance.
- *Def-concept-mapping*: Generates the mappings, specified with the *maps-to* relation, between two ontological concepts.
- *Def-relation-mapping*: Generates a mapping between two relations using a rule definition within an ontology. As OCML represents concept attributes as relations, this primitive can be used to map between input and output descriptions.

22.2.2.4 METEOR-S

The METEOR-S project [76] was carried out at the LSDIS Laboratory at the University of Georgia, with follow on work at the Kno.e.sis Center at Wright State University. METEOR-S supports and leverages semantics through the complete life cycle of Semantic Web processes, encompassing the annotation, publication, discovery, dynamic binding or composition, and enactment of Web Services. The distinguishing characteristic of the research undertaken in the METEOR-S project is the strong coupling with existing Web Services standards [76]. In fact, the philosophy of METEOR-S is to incrementally extend preexisting standards with semantics so as to better support the discovery, composition, and enactment of Web Services. This contrasts with the top-down approaches based on OWL-S and WSMO.

The METEOR-S project has tackled the semantic annotation of Web Services, the semantics-based discovery of Web Services and their composition, which also encompasses data mediation. The remainder of this section will focus on the specific approaches adopted in METEOR-S for each of these research topics. First, the METEOR-S Web Service Annotation

Framework (MWSAF) [77], which contributed to a large extent to the definition of WSDL-S is presented. Second, the focus is on the METEOR-S Web Service Discovery Infrastructure (MWSDI) [78]. Next, the METEOR-S approach to data mediation [79] is described and finally the METEOR-S Web Service Composition Framework (MWSCF) is characterized [80].

Service Annotation

At the center of METEOR-S project is MWSAF [77], a framework for the semiautomatic annotation of Web Services. These annotations address four different aspects of Web Services' semantics. First of all, MWSAF supports the inclusion of annotations about the semantics of the inputs and the outputs of Web Services. Secondly, the annotation framework supports the definition of functional semantics, that is, what the service does. Thirdly, MWSAF enables the inclusion of execution semantics to support verifying the correctness of the execution of Web Services. Finally, the framework incorporates information regarding the quality of service, such as performance or costs associated to the execution of Web Services.

Initial research on the framework was devoted to supporting the semiautomatic annotation of XML Schemas part of Web Services definitions. This work is based on the transformation of both XML Schemas and ontologies into a common representation format called SchemaGraph [77] in order to facilitate the matching between both models. Once the Ontologies and XML Schema are translated into this common representation, a set of matching algorithms can be applied to (semi) automatically enhance the syntactic definitions with semantic annotations.

In a nutshell, the matching algorithm computes a *match score* between each element of the WSDL SchemaGraph and the ontology SchemaGraph. This score takes into account the linguistic and the structural similarity. After all the match scores have been computed, the “best” matching element is chosen by taking into account both the match score and the specificity of the concepts. Finally, a global matching average is computed to help in selecting the best overall match between the Web Services and ontologies. Further details about the algorithm can be found in [77].

MWSAF is composed of three main components: an ontology store, the matcher library, and a translator library. The first component stores the ontologies that will be used for annotating the Web Services. The matcher library provides different algorithm implementations for linguistic and structural matching between concepts and Web Services elements. Finally, the translator library consists of the programs used for generating the SchemaGraph representation for ontologies and Web Services.

MWSAF assists users in annotating Web Services by browsing and computing the concordance between domain models and the Web Service elements. The last step in the annotation process is their representation for future reuse and automated processing. To cater for this the METEOR-S project makes use of SAWSDL, which was presented in

➤ [Section “WSDL-S and SAWSDL.”](#)

Matchmaking

UDDI and the Universal Business Registry (UBR) are the main industrial efforts supporting the automation of Web Services matchmaking. The METEOR-S Web Services

Discovery Infrastructure (MWSDI) attempts to enhance existing Web Services discovery infrastructure by using semantics [78]. MWSDI is a scalable infrastructure for the semantics-based publication and discovery of Web Services.

MWSDI aims to provide unified access to a large number of third-party registries. Thus, in order to provide a scalable and flexible infrastructure it has been implemented using peer-to-peer (P2P) computing techniques. It is based on a four-layered architecture, which includes a Data layer, a Communications layer, an Operator Services layer, and a Semantic Specification layer. The Data layer consists of the Web Services registries and is based on UDDI. The Communications layer is the P2P infrastructure, which is based on JXTA. The Operator Services layer provides the semantic discovery and publication of Web Services. Finally, the Semantic Specification layer enhances the framework with semantics.

MWSDI uses semantics for two purposes. First, it uses *Registries Ontology*, which stores registries information, maintains relationships between domains within MWSDI, and associates registries to them. This ontology stores mappings between registries and domains so that finding Web Services for a specific domain can be directed to the appropriate registries. Additionally, the *Registries Ontology* captures relationships between registries so that searches can be made more selective on the basis of these relationships.

Secondly, MWSDI envisions including domain-specific ontologies for registries, so that Web Services can be annotated by mapping inputs and outputs to existing domain ontologies. The purpose of defining these mappings is to enable semantic discovery by allowing users to express their requirements as *Service Templates*, which are expressed using concepts from the same ontology.

The semantic publication of services in MWSDI registries uses UDDI *tModels* for registering the domain ontologies and *CategoryBags* for categorizing WSDL entities according to one or more *tModels*. MWSDI provides both a manual and a semiautomatic mechanism for defining the mappings between WSDL elements and the concepts in the domain ontologies [78].

Data Mediation

The METEOR-S data mediation technique introduced the *lifting* and *lowering* mechanism for transforming from one representation to another. An overview of Service heterogeneities and the mediation challenges are outlined in [▶ Table 22.1](#). The *lifting* mapping indicates the conversion from the existing format to the common model whereas *lowering* mapping indicates the reverse conversion. An extensive discussion of the mediation mechanism is available in [81].

Composition

Semantic Composition of Web Services in METEOR-S is supported by the METEOR-S Web Service Composition Framework (MWSCF) [80]. In a nutshell, the composition framework aims to increase the flexibility of Web Services composition by making use of *Semantic Process Templates*. *Semantic Process Templates* define processes in terms of

Table 22.1

An outline of service heterogeneities and potential methods to overcome them

Heterogeneities/ conflicts	Examples – conflicted elements shown in color	Suggestions/issues in resolving heterogeneities
<i>Domain incompatibilities</i> – attribute level differences that arise because of using different descriptions for semantically similar attributes		
<p>Naming conflicts Two attributes that are semantically alike might have different names (synonyms) Two attributes that are semantically unrelated might have the same names (homonyms)</p>	<p>Web Service 1 Student(Id#, Name) Web Service 1 StudentId#, Name)</p>	<p>Web Service 2 Student (SSN# Name) Web Service 2 Book(Id#, Name)</p> <p>A semantic annotation on the entities and attributes (provided by <i>WSDL-S:modelReferences</i>) will indicate their semantic similarities</p>
<p>Data representation conflicts Two attributes that are semantically similar might have different datatypes or representations</p>	<p>Web Service 1 Student(Id#, Name) Id# defined as a four digit number</p>	<p>Web Service 2 Student (Id#, Name) Id# defined as a nine digit number</p> <p>*Mapping WS2 Id# is easy with some additional context information while mapping in the reverse direction is most likely not possible</p>
<p>Data scaling conflicts Two attributes that are semantically similar might be represented using different precisions</p>	<p>Web Service 1 Marks 1–100</p>	<p>Web Service 2 Grades A–F</p> <p>*Mapping WS1 Marks to WS1 Grades is easy with some additional context information while mapping in the reverse direction is most likely not possible</p>
<i>Entity definition</i> – entity level differences that arise because of using different descriptions for semantically similar entities		
<p>Naming conflicts Semantically alike entities might have different names (synonyms) Semantically unrelated entities might have the same (homonyms)</p>	<p>Web Service 1 Employee (Id#, Name) Web Service 1 TICKET (TicketNo MovieName)</p>	<p>Web Service 2 Worker (Id#, Name) Web Service 2 TICKET (FlightNo.Arr, Airport,Dep, Airport)</p> <p>A semantic annotation on the entities and attributes (provided by <i>WSDL-S:modelReferences</i>) will indicate their semantic similarities</p>
<p>Schema isomorphism conflicts Semantically similar entities may have different number of attributes</p>	<p>Web Service 1 PERSON (Name, Address, HomePhone, WorkPhone)</p>	<p>Web Service 2 PERSON (Name, Address, Phone)</p> <p>*Mapping in both directions will require some additional context information</p>

Table 22.1 (Continued)

Heterogeneities/ conflicts	Examples – conflicted elements shown in color		Suggestions/issues in resolving heterogeneities
<i>Abstraction level incompatibility</i> – entity and attribute level differences that arise because two semantically similar entities or attributes are represented at different levels of abstraction			
Generalization conflicts Semantically similar entities are represented at different levels of generalization in two Web Services	Web Service 1 GRAD-STUDENT, (ID, Name, Major)	Web Service 2 STUDENT(ID, Name, Major, Type)	*WS2 defines the student entity at a much general level. A mapping from WS1 to WS2 requires adding a Type element with a default “Graduate” value, while mapping in the other direction is a partial function
Aggregation conflicts Semantically similar entities are represented at different levels of generalization in two Web Services	Web Service 1 PROFESSOR(ID, ProfID, Dept)	Web Service 2 FACULTY (ID, Name, Dept)	*A set of Professor entities is a Faculty entity. When the output of WS1 is a Professor entity, it is possible to identify the Faculty group it belongs to, but generating a mapping in the other direction is not possible
Attribute entity conflicts Semantically similar entity modeled as an attribute in one service and as an entity in the other	Web Service 1 COURSE (ID,Name, Semester)	Web Service 2 DEPT(Course sem, ,)	*Course modeled as an entity by WS1 is modeled as an attribute by WS2. With definition contexts, mappings can be specified in both directions

*Interoperation between services needs transformation rules (mapping) in addition to annotation of the entities and/or attributes indicating their semantic similarity (matching)

semantically defined activities. Using these *Semantic Process Templates*, executable processes can be generated by binding the semantically defined activities to concrete Web Services that conform to the activity specification.

MWSCF is composed of four components: the process builder, the discovery infrastructure (see [Sect. Matchmaking](#)), XML repositories, and the process execution engine. The process builder includes a graphical user interface for defining Semantic Process Templates and a process generator. The process generator retrieves ontologies, activity interfaces, and process templates from the XML repositories and uses MWSDI for discovering suitable Web Services, in order to transform the templates into executable processes. The executable process definitions can then be handed to the process execution engine for the actual execution of the Web Services composition.


In MWSCF, *Semantic Process Templates* [82] are basically a set of Activities connected by means of Business Process Execution Language (BPEL) [18] control-flow constructs. Activities can be defined with a varying degree of flexibility by using a specific Web Service implementation, a Web service interface, or a *Semantic Activity Template*. Specific Web Service implementations can be specified for static compositions. Web Service interfaces can be applied to gain some flexibility allowing diverse implementations of the same interface to be interchangeably executed. Finally, *Semantic Activity Templates* provide a greater degree of flexibility by defining activities semantically in terms of their inputs, outputs, and functional semantics, for example, preconditions and effects.

The creation of an executable process is a semiautomated activity performed at design-time where the user is assisted in refining the template with concrete Web Services and data-flow. In order to do so, Web Services that implement the specified Web Service interfaces are retrieved from the XML Repository and the MWSDI is used for discovering suitable services when *Semantic Activity Templates* have been specified. After all the activities have been replaced by concrete Web Services, the user can map Web Service outputs to other service inputs in order to define the process data-flow. Once the explicit data-flow has been defined, the process generator creates the executable representation, which is a BPEL4WS process that can be executed in any BPEL execution engine.

METEOR-S research also addressed dynamic process composition [82], WS-agreement-based partner selection [83] optimal process adaptation with constraints [84], and event identification [85].

22.3 Example Applications

So far, the main principles underlying SWS have been discussed along with the main conceptual models devised so far and the main frameworks able to deal SWS descriptions to achieve some automation for some of the tasks involved in the life cycle of Web Services. In this section, a couple of examples illustrating some of the features provided by SWS and also showing how SWS applications can be developed are provided. This section is mostly for explanatory purposes and is therefore not to be taken as a proof of the capabilities of SWS nor as a recipe for constructing applications since this would require a level of detail and complexity that is beyond the scope of this chapter.

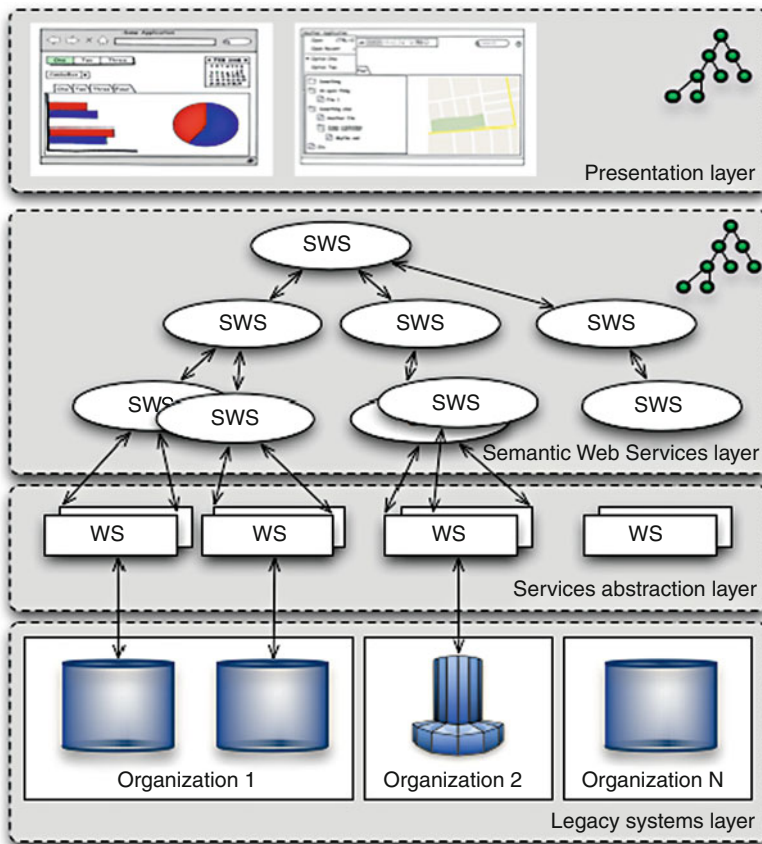
Although applications always present particular architectural characteristics and constraints inherited from the environment where they are deployed or the concrete requirements that need to be fulfilled, SWS applications typically are structured around the following four layers shown in  Fig. 22.6:

- *Legacy system layer*: Consists of the existing data sources and IT systems available from each of the organizations involved in the integrated application.
- *Service abstraction layer*: Exposes (micro-) functionalities of the legacy systems as Web Services, abstracting from the hardware and software platforms.

- *Semantic Web Service layer*: To set up an application, a set of application-specific SWS descriptions has to be provided. These descriptions are typically centrally stored within some repository for easy querying and retrieval through a more or less advanced discovery or service matchmaking machinery.
- *Presentation layer*: Consists of a Web application accessible through a standard Web browser. The possible actions presented depend on the underlying services available. Typically, input and output data are directly gathered from and presented to the user based on some semantic representation such as RDF(S), OWL, or WSML.

22.3.1 Applying Semantic Web Services in eGovernment

In the context of the European project DIP (<http://dip.semanticweb.org/>), a close collaboration was established with the Essex County Council (ECC) – a large local authority



■ Fig. 22.6

Typical architecture of Semantic Web Services (SWS)-based applications

in South East England (UK) comprising 13 boroughs and containing a population of 1.3M – to deploy real-world applications in the eGovernment domain based on SWS technologies [70, 86]. During this collaboration, a framework designed around IRS-III was developed, tested, and refined. The use cases show how SWS technology provides an infrastructure in which new services can be added, discovered, and composed continually, and the organization processes automatically updated to reflect new forms of cooperation.

This section illustrates the development and application of SWS by describing two compelling use cases in the eGovernment domain: Change of Circumstances and the Emergency Management System. The first application illustrates how Web Services and ontologies can be leveraged in order to support a seamless integration and interaction within and between governmental administrative organizations to automatically handle the change of a citizen situation. In the second one, the developed application supports emergency planning and management personnel by retrieving, filtering, and presenting data from a variety of legacy systems to deal with a specified hazardous situation. This second scenario puts more emphasis on the dynamics of SWS technologies, illustrating how systems can dynamically and transparently choose and orchestrate specific services in order to better deal with the situation at hand. Although the two examples are based on WSMO and IRS-III as the specific execution environments, most of the techniques and solutions illustrated herein could be supported using different modeling and execution frameworks.

22.3.1.1 Change of Circumstances

Tiers of government – national, county, and district – largely operate autonomously, without central control of service provision. Additionally, each tier has distinct viewpoints that may differ from that of general citizens. Therefore, integration and interoperability are significant requirements in the development of applications in the eGovernment domain encompassing diverse administrative bodies.

Integration requires the assembly and transformation of processes needed to support specific user tasks into a single service with the corresponding back-office practices by integrating multiple governmental entities at different levels. Interoperability is a key issue in order to allow for data and information to be exchanged and processed seamlessly across the agencies involved. Interoperability is not only a technical issue but also a fundamental requirement to share and reuse knowledge between networks and administrations, which often calls for the reorganization of administrative processes. Interoperability problems therefore span technical issues such as interfaces, data formats, and protocols; semantic issues concerning the exchange of information in an understandable way; and finally organizational issues regarding the modeling of business processes suited to how governmental agencies work internally.

The application was developed to solve a specific use case problem at Essex County Council (ECC). Whenever the circumstances in which a given citizen lives change, he/she might be eligible for a set of services and benefits provided by ECC and other governmental agencies together with public service providers. An example of such a change of

circumstances is, if an elderly, partly disabled woman moves in together with her daughter. This is a change of circumstances for both, the mother and the daughter. For instance, the mother might no longer receive a “meals-on-wheels” service, whereas the daughter might get financial supporting for caring her mother. Starting from existing legacy systems, the aim is to provide integrated functionalities such as change of patient details within multiple legacy systems, change of patient pending equipment orders, a list of all services for a patient, the cessation of some services, and patient equipment assessment.

Generally, even very simple processes in a change of circumstances require the interaction of many different government agencies. Each agency has different legacy systems in place to keep track of citizen records, provide services, third-party service providers, etc. In the application discussed herein, the following two data sources provided by two different departments (at two distinct governmental levels) were considered:

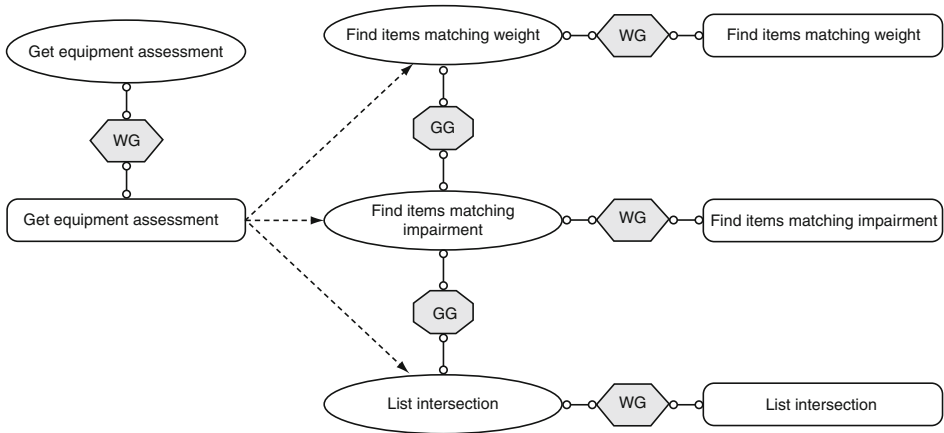
- *Citizen assessment (community care department of the ECC)*: This relates to information about citizens registered in ECC for assessment of services and benefits (e.g., meals-on-wheels; house cleaning; and in situ patient car). This information is stored in the SWIFT database.
- *Order equipment (housing department of the Chelmsford district council)*: This relates to information about equipment (e.g., stair lift, wheel chair, and crutch), which is provided to citizens registered in Essex. This information is stored in the ELMS database.

On top of the two legacy systems, a set of Web Services that perform SQL queries and carried out certain basic operations for dealing with changes of circumstances were developed. In particular, 8 Web Services that interact with the SWIFT database and 19 Web Services interacting with the ELMS database were created.

A number of ontologies were defined covering the whole application domain and also providing the SWS descriptions required to achieve the goals. In particular, three domain ontologies were integrated with two ontologies specific for each of the back-end systems, and other three including the SWS descriptions which, being based on WSMO (see [▶ Sect. “WSMO”](#)) therefore comprises Goals, Web Services, and Mediators definitions.

Each ellipse in [▶ Fig. 22.7](#) represents a Goal that has to be accomplished by simple or integrated services – represented as rectangles. Specifically, the three goals in the middle are accomplished by functionalities provided by Web Services available. Such goals are automatically orchestrated to accomplish the main goal on the left, namely *Get Equipment Assessment*. The goal defines two inputs (weight and impairment) and one output (equipment list) expressed in terms of the domain ontologies for the application. At runtime – when the goal is invoked to be accomplished – the instances of the input classes are selected through the user interface of the application, while the result instances of the catalog-data class are created on the fly by lifting them from the syntactic representation obtained from the results of Web Service invocations.

Each Goal includes a set of Web Service(s) that can achieve the required objectives. In this case, there is only one Web Service per Goal so there is no need to apply any advanced matchmaking functionalities in order to choose the most suitable Web Service to use (the next use case illustrates how these situations can be handled). Additionally, as explained



■ Fig. 22.7

Structure of the SWS descriptions created for the Find Item ELMS by impairment and weight functionality

earlier in [Sect. 22.2.2.3](#), one WG-Mediator connects every Goal to appropriate Web Services and takes care of data transformations between heterogeneous ontologies if necessary.

The Goal *Get Equipment Assessment*, shown in [Fig. 22.7](#), is a composite Goal whose objective is achieved by orchestrating three subgoals. The actual orchestration definition for this example is a simple sequence as illustrated in [Listing 22.10](#). Although the definition described in the listing is in OCML and based on the WSMO conceptual model, the reader should note that a similar approach could be followed in OWL-S by defining a Service Process Model or using BPEL. Each subgoal, invoked through the orchestration, is conversely directly accomplished by invoking the respective Web Service.

Listing 22.10: Get equipment assessment goal definition

```
(DEF-CLASS GET-EQUIPMENT-ASSESSMENT (GOAL) ?GOAL
  ( (HAS-INPUT-ROLE :VALUE HAS-CITIZEN-WEIGHT
      :VALUE HAS-CITIZEN-DISEASE
      :VALUE HAS-CASE-WORKER-CODE)
    (HAS-OUTPUT-ROLE:VALUE HAS-SUITABLE-ITEMS-LIST)
    (HAS-CITIZEN-WEIGHT :TYPE NUMBER)
    (HAS-CITIZEN-DISEASE :TYPE DISEASE)
    (HAS-CASE-WORKER-CODE :TYPE NUMBER)
    (HAS-SUITABLE-ITEM-LIST :TYPE ITEM-LIST) ) )
(DEF-CLASS GET-EQUIPMENT-ASSESSMENT-INTERFACE-ORCHESTRATION
  ( (HAS-BODY
    :VALUE ( (ORCH-SEQUENCE
```

```

GET-EQUIPMENT-ASSESSMENT-GOAL
CHECK-EQUIPMENT-CASE-WORKER-GOAL)
(ORCH-RETURN (ORCH-GET-GOAL-VALUE CHECK-EQUIPMENT-CASE-
WORKER-GOAL) ) ) ) )

```

This example application has illustrated how one can define simple Semantic Web Services by means of WSMO to implement cross-organizational integrated functionality, abstracting from the underlying legacy systems, keeping the autonomy of involved parties, and covering multiple heterogeneous domains. If new systems need to be integrated, one can simply introduce the appropriate SWS descriptions and mediation facilities when mismatches occur. This example application contrasts with traditional database and Web Services techniques that would necessitate that the different parties involved harmonize their database schemas and agree upon concrete service interfaces. The next application illustrates a slightly more complex use of SWS technologies that shows how dynamic Web Service matchmaking can provide additional flexibility to systems.

22.3.1.2 eMerges

In an emergency, multiple agencies need to collaborate, sharing data and information about actions to be performed. However, many emergency-relevant resources are not available on the network and interactions among agencies or emergency corps usually occur on a personal/phone/fax basis. The resulting interaction is therefore limited in scope and slower in response time, contrary to the nature of the need for information access in an emergency.

Emergency-relevant data are often spatial-related. *Spatial-Related Data (SRD)* is traditionally managed with the help of *Geographical Information Systems (GIS)*, which allow access to different layers of SRD such as highways, transportation, postal addresses index, and land use. GIS support decision-making by facilitating the integration, storage, querying, analysis, modeling, reporting, and mapping of these data.

The prototype, called eMerges [86], is in effect a decision support system, which assists the end user – currently the Emergency Planning Officer – in assembling information related to a certain type of event, more quickly and accurately. The application's user interface, shown in [Fig. 22.8](#), is based on Web standards. XHTML and CSS are used for presentation, while JavaScript is used to handle user interaction together with AJAX techniques to communicate with IRS-III. One of the main components of the interface is a map, which uses the Google Maps API to display polygons and objects (custom images) at specific coordinates and zoom level. Each time an object is displayed by a user at a particular location, the user interface provides a set of contextually relevant actions that can be carried out and the corresponding invocation form, should the user wish to invoke a particular action.

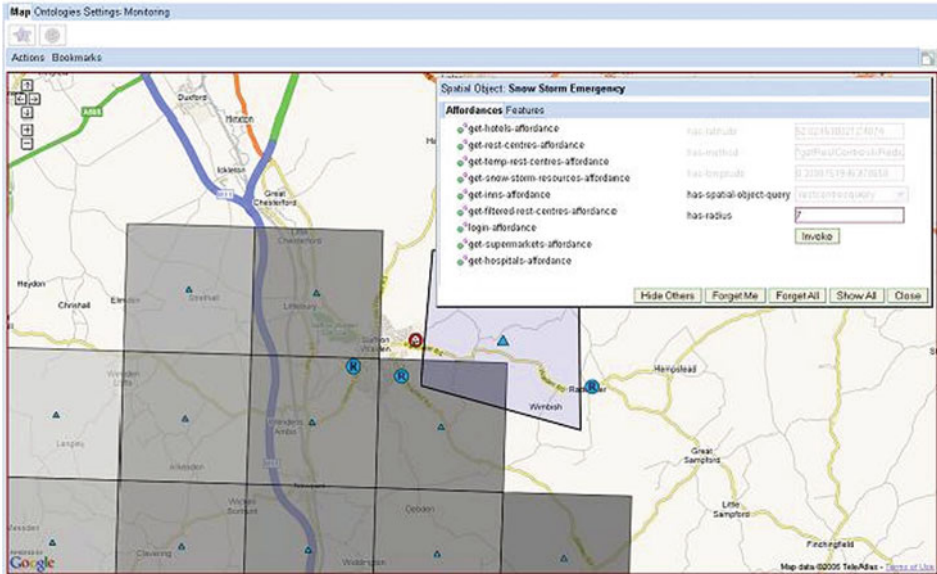


Fig. 22.8
Screenshot of eMerges

A number of ontologies were defined covering the application domain and also providing the SWS descriptions required to support emergency officers in handling extreme situations. In particular, three service-oriented ontologies were developed for describing the domains of the services provided by the back-end systems. These ontologies were integrated with user-oriented domain ontologies providing concepts for describing spatial aspects and context among other things. Finally, three ontologies captured the SWS descriptions, that is, the Goals, Web Services, and Mediators, for meteorological information SWS, emergency planning SWS, and for dealing with distributed teams through a messaging system.

The purpose of the application ontologies is the aggregation of different data sources on, respectively, a representation, a cognitive level, and a spatial level. They allow the different data sources to be handled and presented in a similar way. Inversely to the lifting operations, *lowering operations* transform instances of aggregation ontologies into syntactic documents to be used by the server and client applications.

The emergency management system aggregates data and functionalities from three different sources:

- *Meteorological office*: Is a national UK organization that provides environmental resources, such as weather forecast, snow, and pollution data.
- *ViewEssex*: Is a collaboration between ECC and British Telecommunications (BT), which has created a single corporate spatial data warehouse. As can be expected

ViewEssex contains a wide range of data including data for roads, administrative boundaries, buildings, and Ordnance survey maps, as well as environmental and social care data.

- *BuddySpace*: Is an Instant Messaging client facilitating lightweight communication, collaboration, and presence management built on top of the instant messaging protocol Jabber (Jabber. <http://www.jabber.org/>). The BuddySpace client can be accessed on standard PCs, as well as on PDAs and mobile phones, which in an emergency may be the only hardware device available.

eMerges distinguishes between two classes of services: *data* and *smart*. The former refer to the three data sources introduced above, and they are exposed by the following standard Web Services:

- *Meteorological services*: Provide weather information, for example, snowfall level over a given rectangular spatial area.
- *ViewEssex services*: Return detailed information on specific types of rest center. For example, *Get Hospitals* is a Web Service that returns a list of relevant hospitals within a given circular area.
- *BuddySpace services*: Allow the presence information of online users to be accessed.

Smart services represent specific emergency planning reasoning and operations on the data provided by the data services. They are implemented in a mixture of Common Lisp and OCML and make use of the emergency management system ontologies. In particular, the application includes a number of services that filter the data retrieved from ViewEssex according to emergency-specific requirements, for example, rest centers with heating system, hotels with at least 40 beds, and easy accessible hospital.

➤ *Figure 22.9* shows an example of the created SWS descriptions: *Get Polygon GIS Data with Filter* represents a request for available shelters within a given area. The user specifies a polygon area and the shelter type (e.g., hospitals, inns, and hotels). The results obtained by querying ViewEssex need to be filtered in order to return shelters correlated to emergency-specific requirements only. This scenario involves (i) *selecting* the appropriate ViewEssex Web service; (ii) *mediating* the difference in area representations (polygon vs. circular) between the user goal and available Web Services; (iii) *orchestrating* the retrieve and filter data operations to eventually achieve the user's goal.

The SWS representations address the Web Service selection problems as follows. When a user wants to achieve the *Get Circle GIS Data* Goal, IRS-III gets all the possible Web Services that can solve it by means of the WG-Mediators. Each semantic description of ViewEssex Web Service defines the Web Service capability (see ➤ *Sect. "WSMO"*), that is, the functionality it provides, based on the class of shelter provided by the Web Service. ➤ *Listing 22.11* reports an example in OCML on how the assumption of a capability can be defined. If the Web Service provides the class of shelters defined in one of the inputs of the goal, IRS-III selects it. In the example above, the Web Service is selected if the request class of shelters are hospitals.

Listing 22.11: Example of a capability definition in OCML

```
(DEF-CLASS GET-ECC-HOSPITALS-WEB-SERVICE-CAPABILITY (CAPABILITY)
?CAPABILITY
( (USED-MEDIATOR :VALUE GET-GIS-DATA-MEDIATOR)
(HAS-ASSUMPTION:VALUE
(KAPPA (?WEB-SERVICE)
(= (WSMO-ROLE-VALUE ?WEB-SERVICE 'HAS-SPATIAL-OBJECT-
QUERY)
'HOSPITALSQUERY) ) ) )
```

This use case therefore provides, as in the previous case, the ability to aggregate and reuse diverse information resources relevant to a given situation in a cost-effective way and to make this available as a basis for transparent interaction between community partner organizations and individual users. Additionally, this use case highlights the following aspects:

- Complex SWS descriptions were created on top of three distinct kinds of legacy system: database, GIS, and instance messaging. The use of Web Services allows one to abstract from the underlying technologies and ease thus their integration.

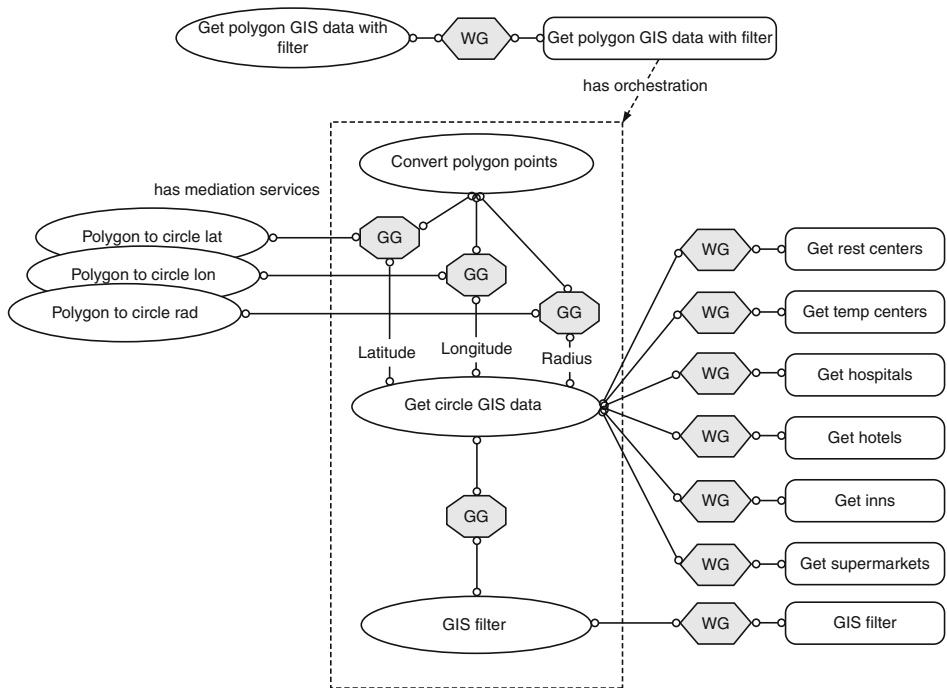


Fig. 22.9

A portion of Web Service Modeling Ontology (WSMO) descriptions for eMerges

- A given Goal, for example, *Get Circle GIS Data*, might be achieved by several Web Services. The most appropriate one is selected on the basis of the specific situation. The actual workflow, that is, the sequence of service invocations, is established at runtime only and it is therefore possible to better adapt the execution to the concrete context and situation at invocation time. In existing Web Service–based approaches, the functionalities are mapped at design-time, when the actual context is not known and therefore one can only reach this level of adaptability by complicating unnecessarily the workflow definition.
- The use of WG- and GG-Mediators allows goal and process mediation and thus a smooth crossing among services of distinct domains in the same workflow. As with the previous case, the appropriate mediation service is selected at runtime, according to the specific situation.
- If new Web Services are integrated, for instance providing data from further GIS, new Web Service descriptions can be simply introduced and linked to the Goals by means of Mediators. In the same way, new filter services, for example, more efficient ones, may be introduced with minimal changes.

22.4 Related Resources and Key Papers

This section briefly revises some of the main papers in the area and also provides a set of related resources that are considered of particular importance for understanding the research results produced up to date. For space reasons, this is a distilled set of resources limited to those that are considered have been more influential in a given area of research within SWS. The resources have been ordered by year of publication.

22.4.1 Key Papers

McIlraith et al. present in [7] what is perhaps the first comprehensive attempt to describe Semantic Web Services, the principles underlying them and the possible approaches for achieving this ambitious vision. The paper suggests how Web Services can be annotated with DAML-S and outlines the main requirements for supporting the automation of Web Services discovery, composition, and execution. Finally, the authors present a proof of concept implementation based on agents' technology. This paper established the main notions underlying Semantic Web Services and subsequently triggered and motivated a good part of the research around SWS described herein.

In [10], Fensel and Bussler introduce the Web Service Modeling Framework (WSMF), a conceptual model for developing and describing Web Services and their compositions, which subsequently gave birth to the Web Service Modeling Ontology (WSMO) and Web Service Execution Environment (WSMX). The distinctive characteristics of WSMF are the two complementary principles it builds upon: an extreme decoupling of components supported by making mediation between components a central aspect. In this paper, the authors outline the core issues related to achieving the vision promoted

by Web Services and outline the main aspects that need to be described in a formal manner to cater for further automation.

In [8, 9], Sheth and Sivashanmugam et al. present the first attempt to SWS based on extending existing SOA standards and introduced the four main types of semantics that Semantic Web Service descriptions can capture. In particular, the authors identify data semantics capturing the data exposed and used by services, functional semantics formalizing the functionality of the service, nonfunctional semantics such as the quality of service, and execution semantics (e.g., execution errors).

In [52], Sycara et al. present in detail the work they have carried out on Semantic Web Services using DAML-S, which then became OWL-S. Their paper describes in detail their approach to service matchmaking as well as the DAML-S Virtual Machine that is able to interpret DAML-S processes and support their execution. Finally, they briefly introduce their work on using DAML-S descriptions for Web Services composition by using an existing planner. The framework described in this paper was probably the most advanced SWS environment at that point on time and is still considered as a reference for subsequent research.

In [11], Preist carries out a thorough analysis of a number of use cases in order to highlight the main requirements for SWS and execution environments. Based on this analysis, Preist presents a conceptual architecture for SWS based largely on the W3C Web Services Architecture, which is extended to cater for the additional features he identifies. The conceptual architecture defined by the author interleaves conceptual aspects that are of central relevance for formalisms for SWS with procedural concerns aimed at highlighting the activities that execution environments need to support and how the conceptual model defined covers these aspects. Much of the work presented in this paper subsequently informed the development of SWS models and execution environments.

In [56], Li and Horrocks present an advanced approach to service matchmaking that treats service advertisements and requests as whole concepts as opposed to most of the approaches devised thus far that treat inputs and outputs; preconditions and effects separately in most cases. Thus, this work provides a somewhat higher level of abstraction closer to the problem faced by developers of service-oriented systems that need activities to be fulfilled rather than specific services fitting certain inputs and outputs schemas. Additionally, this approach has also the merit of being entirely based on Description Logics, which therefore makes it particularly suitable for existing Semantic Web languages.

In [61], Traverso and Pistore propose a planning technique for supporting the automated composition of services described in OWL-S. The authors propose an approach that is able to deal with nondeterminism, partial observability, as well as complex goals and generates executable compositions in BPEL4WS. Their planner makes use of semantic annotations in order to enhance the performance of the composition processes as compared to automating the composition by dealing directly with the syntactic descriptions of processes. One fundamental outcome of this research is precisely the fact that they illustrate how the use of semantic annotations can help improve the performance of Web Service composition, therefore showing some of the potential benefits semantics can bring to Web Services.

In [21], Martin et al. describe OWL-S, formerly DAML-S, one of the main ontologies for the description of Semantic Web Services expressed in OWL. OWL-S defines an upper ontology for semantically describing Web Services along their *Profile*, that is, “what the service does,” their *Model*, that is, “how the service works,” and their *Grounding*, that is, “how the service can be accessed.”

In [12], Burstein et al. describe the main outcomes of the Semantic Web Services Initiative Architecture committee, which are a set of architectural and protocol abstractions that define the foundations for Semantic Web Service technologies. This work although essentially in line with prior work on conceptual models and architectures like [11] and [10], provides a more agent-oriented representation defining an interoperability model that can underpin diverse implementations.

The book edited by Cardoso and Sheth [87] is a quite extensive compilation of diverse work carried out in Semantic Web Services. This book presents some of the different approaches to annotating Semantic Web Services. It also tackles aspects such as the matchmaking of services, the choreography of services, and the use of temporal reasoning for supporting reactive processes. Finally, the book presents a number of real-world applications that exploit Semantic Web Services technologies thus illustrating but also motivating the techniques described. Overall, this book presents a very valuable overview of some of the main aspects around Semantic Web Services and therefore complements this chapter with more detailed information about some of the topics exposed herein.

In [23], Farrell and Lausen provide the specification of SAWSDL, which is, at the time of this writing, the only Web standard for Semantic Web Services. SAWSDL is a lightweight bottom-up approach to bringing semantics to Web Services described in WSDL and provides also a means for annotating XML Schema. The specification identifies three kinds of hooks, namely *Model Reference*, *Lifting Schema Mapping*, and *Lowering Schema Mapping*. The former provides means for linking certain WSDL and XML elements to semantic elements through URIs. The other two on the other hand provide the means for specifying how syntactic data are transformed into their semantic counterpart and vice versa.

In [19], Fensel et al. compile a good deal of the body of research carried out around WSMO. The book therefore describes the ontology in detail explaining the rationale behind its core elements Goal, Web Service, Mediator, and Ontology and how they help to support the automation of typical activities involved when using Web Services. The book additionally provides details on the Web Service Modeling Language (WSML) that supports describing WSMO entities and provides examples on how these descriptions can be utilized to support the discovery, composition, and execution of services among others.

Providing a more architectural perspective over the construction of systems that can make use of Semantic Web Services descriptions to cover the entire life cycle of SWS-based applications, [15] complements [19]. The book presents a conceptual architecture for Semantic Execution Environments and provides details for how these components shall be integrated and how they internally achieve their particular goals.

In [70], Domingue et al. thoroughly describe IRS-III one of the main frameworks for SWS, which is covered in [▶ Sect. 22.2.2.3](#). IRS-III is a broker-based platform based on research on Problem-Solving Methods that mediates between clients and service providers

allowing each of them to adopt the most convenient means for representing their perspective, while supporting an effective interaction. It is largely based on WSMO and provides support for ranking and selecting services, as well as orchestrating and invoking them by carrying out the appropriate data mediation and choreography of message exchanges.

In [44], Sheth et al. present SA-REST an open, flexible, and standards-based approach to adding semantic annotations to RESTful services and Web APIs. SA-REST is perhaps the first serious approach to tackling the annotation of the emerging RESTful services and Web APIs thus bringing them into the Semantic Web Services spectrum. SA-REST uses GRDDL and RDFa as a means to structure Web pages and support the embedding of semantic annotations within them. On top of these microformats, SA-REST supports the linking of service descriptions to semantic metamodels by using model reference type of annotations from SAWSDL. The use of SA-REST and semantically annotated Web APIs for rapidly creating smart or semantic Smashups and for supporting semantic search and ranking of Web Services are discussed in [47] and [88], respectively.

WSMO-Lite [38] is a recent approach to bring semantics to SAWSDL. WSMO-Lite identifies four main types of semantic annotations for services, namely functional semantics, nonfunctional semantics, behavioral semantics, and the information model and how these can be expressed by means of SAWSDL annotations. Additionally, WSMO-Lite provides a simple RDF Schema that allows one to refine the semantics of the SAWSDL links in an incremental and compatible manner.

22.4.2 Related Papers

The CommonKADS methodology for constructing Knowledge-Based Systems is described in [26]. This book accounts for a large body of research carried out in the context of Problem-Solving Methods, which aims at providing systematic means including a conceptual framework as well as a methodology for the development of systems that solve knowledge intensive tasks by reusing general purpose yet task-specific components. Although the notion of service is not directly contemplated in this research, certain approaches to Semantic Web Services notably WSMO and IRS-III are largely based on many of the notions from research on Problem-Solving Methods that this book addresses.

The specifications for WSDL 1.1 and WSDL 2.0 are provided in [89, 90] respectively. Although WSDL solely provides syntactic means for describing Web Services, it is one of the most widely applied technologies for describing services on the Web and indeed most of the approaches to describing services semantically are compatible with it.

The initial specification of the Business Process Execution Language (BPEL) and subsequent refinement are provided in [18, 91]. BPEL is the de facto standard for the specification of executable compositions of Web Services in the industry and has therefore been utilized within Semantic Web Service applications to support the orchestration of services. Some researchers have additionally worked on providing semantic extensions [61, 68, 92].

In [93], Staab et al. present an overview of the main issues faced by Web Services both from an industrial perspective as well as from the standpoint of researchers focusing on Semantic Web Services. This series of short articles presents a quick yet remarkable comparison of existing process modeling languages, highlighting their features and drawbacks. Additionally, it also covers conceptual concerns that highlight the lessons that can be learnt from prior work on Problem-Solving Methods, as well as conceptual models for describing services semantically. Overall, it provides a brief yet valuable overview of issues and approaches that can help better understand the research and development carried out so far in the area of Web Services and Semantic Web Services.

Most of the research on attaching semantics to services has focused mainly on the technical aspects. In [64], Akkermans et al. provide a different perspective focusing on the essential features of services from a business standpoint. This research, together with other complementary work by the authors on a suite of business-oriented ontologies and tools, is complementary to that focusing on technical aspects. It provides an advanced framework for the development of business solutions that are able to adequately fulfill customer needs and support the analysis and combination of business services into added-value solutions.

A few of the growing body of specifications defined for Web Services are included in [35, 94–98]. These specifications cover aspects such as service registries, the brokering of messages through queues, the definition of policies, and the establishment of trustworthy and secure communications. Listing them all is out of the scope of this chapter; however, the reader should note that these new specifications are likely to trigger the need for semantic enrichment and alignment with initiatives coming from the Semantic Web.

22.5 Future Issues

After almost a decade, research on SWS has produced a wealth of conceptual models, languages, architectures, algorithms, and engines that highlight the potential of these technologies both for enterprise settings and for the Web. This chapter has introduced some of the main results and has also provided examples of applications that have explored and showcased the use of SWS in real-world settings. Despite the advances, however, the vision initially proposed in [6] and later on highlighted in [7] when SWS were first proposed is still to be achieved. In order to outline what could be the future issues to be addressed and what the future could bring to this field, it is therefore necessary to analyze the current situation and try to identify what the reasons for this reduced take-up are. An example of critical analysis appears in [99].

Research on SWS so far has focused mostly on extending existing Web Services technologies with semantics. Yet, recent analyses estimate that the number of publicly available Web Services on the Web is around 27,000, which contrasts with the number of Web pages available and with the number of services that big companies have internally (e.g., approximately 1,500 for Verizon) [4]. Hence, despite their name, Web Services seem

to be essentially enclosed within enterprises and it has been argued that indeed Web Services are not really thought for the Web [100]. The application of SWS technologies mostly concerns enterprises and in this respect, the appearance of SAWSDL as the first W3C standard for annotating Web Services had a positive impact in the take-up of SWS technologies. On the Web, the use of SWS is even scarcer and it seems that the appearance of intelligent Web agents that act on behalf of users remains an elusive target. Still, the demand for services on the Web exists as indicated by the proliferation and popularity of publicly available Web APIs and RESTful services.

Additionally, SWS are particularly demanding from a knowledge acquisition perspective. Creating a rich semantic description of a Web Service requires the use of domain ontologies, the use of services taxonomies, the definition of lifting and lowering mechanisms, and in some cases the inclusion of complicated logical expressions. This tedious and complex annotation process has arguably hampered the adoption of SWS technologies especially in the early days of the Semantic Web when publicly available ontologies and semantic information were scarce. As a consequence, the application of SWS technologies within real applications is usually limited to simple annotations that simplify the retrieval of services. This leaves aside more advanced features such as the automated selection of services and therefore reduces the potential benefit that can be obtained.

In the short term, the future of services will continue and exacerbate current trends that are giving birth to a dichotomy between services for the enterprise (i.e., Web Services and the WS-* stack) and services for the Web (i.e., Web APIs). In the long run, with the wider use of services on the Web, it is expected that both technologies will gradually fuse driven by the interest of companies to also address the long tail of customers on the Web. The future of SWS research is undeniably going to be closely related to the evolution of services technologies. Consequently, research on SWS will be affected by this evolution and there will be two main application areas, one driven by the use of services in the enterprise and the other by the application of services on the Web.

One application area will therefore focus on providing semantic annotations to the growing body of WS-* standards in order to properly describe, communicate, and reason about processes, policies, trust, and security. This area will work on addressing some of the main research challenges in the area such as the self-management, self-adaptation, and the governance of service-oriented systems, aspects that have indeed traditionally been of concern for SWS [101]. The other application area will bring in, extend, and adapt the results gathered so far, to deal with the intricacies of Web APIs. In this area, major efforts will have to be devoted to adequately dealing with an open environment like the Web, thus truly assuming its heterogeneity and dynamicity but also exploiting extensively the benefits that can be obtained by using semantics. Also, the fact that Web applications are turning toward handcrafted Web APIs rather than WSDL Web Services will drive the current focus of Semantic Web Services in this second area of application putting the Web back at the center of the research.

Alongside these major trends, impelled by the technical evolution of services, research on SWS needs to focus on the main pending issue: its take-up. This will necessarily have to put the emphasis on reducing the annotation bottleneck and on properly accommodating

Web trends above other aspects. One shall see further solutions based on lightweight annotations (e.g., SAWSDL and WSMO-Lite) as has happened in the Semantic Web in general. Still, SWS will necessarily have to move toward a consensus on the languages used and possibly SAWSDL and RDF(S) are good starting points.

Additionally, work on services will need to integrate with the linked data phenomenon. Linked data will, on the one hand, provide a wealth of data able to reduce to a certain extent the semantic annotation bottleneck. On the other hand, the linked data community and the Semantic Web in general will require the development of more and more complex distributed solutions for which services are an adequate software engineering abstraction. In essence, a significant step toward a greater adoption and use of SWS technologies lies in viewing Semantic Web Services as services for the Semantic Web rather than as semantics on top of Web Services.

22.6 Cross-References

- [eBusiness](#)
- [KR and Reasoning on the Semantic Web: OWL](#)
- [KR and Reasoning on the Semantic Web: RIF](#)
- [Ontologies and the Semantic Web](#)
- [Semantic Annotation and Retrieval: RDF](#)
- [Semantic Annotation and Retrieval: Web of Data](#)
- [Semantic Annotation and Retrieval: Web of Hypertext – RDFa and Microformats](#)
- [Semantic Web Architecture](#)

References

1. Erl, T.: SOA Principles of Service Design. Prentice Hall, Boston (2007)
2. Papazoglou, M.: Web Services: Principles and Technology. Prentice Hall, Boston (2007)
3. Van Der Aalst, W.: Don't go with the flow: web services composition standards exposed. *IEEE Intell. Syst.* **18**, 72–76 (2003)
4. Davies, J., Domingue, J., Pedrinaci, C., Fensel, D., Gonzalez-Cabero, R., Potter, M., Richardson, M., Stincic, S.: Towards the open service web. *BT Technol. J.* **26**, 2 (2009)
5. Fielding, R.T.: Architectural styles and the design of network-based software architectures. Doctoral dissertation, University of California, Irvine (2000)
6. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. *Sci. Am.* **284**(5), 34–43 (2001)
7. McIlraith, S., Son, T., Zeng, H.: Semantic web services. *IEEE Intell. Syst.* **16**, 46–53 (2001)
8. Sheth, A.: Semantic Web Process Lifecycle: Role of semantics in annotation, discovery, composition and orchestration. In: WWW 2003 Workshop on E. Services and the Semantic Web (invited talk), Budapest (2003)
9. Sivashanmugam, K., Verma, K., Sheth, A., Miller, J.: Adding semantics to web services standards. In: Proceedings of the 2003 International Conference on Web Services (ICWS 2003), Las Vegas, pp. 395–401 (2003)
10. Fensel, D., Bussler, C.: The web service modeling framework WSMF. *Electron. Commer. Res. Appl.* **1**, 113–137 (2002)
11. Preist, C.: A conceptual architecture for semantic web services. In: Proceedings of the Third International Semantic Web Conference (ISWC 2004), Hiroshima. Lecture Notes in Computer Science, vol. 3298, pp. 395–409. Springer, Berlin (2004)

12. Burstein, M., Bussler, C., Zaremba, M., Finin, T., Huhns, M.N., Paolucci, M., Sheth, A.P., Williams, S.: A semantic web services architecture. *IEEE Internet Comput.* **9**, 72–81 (2005)
13. Norton, B., Pedrinaci, C., Domingue, J., Zaremba, M.: Semantic execution environments for semantics-enabled SOA. *IT – methods and applications of informatics and information technology. Special Issue in Service-Oriented Arch.*, pp. 118–121 (2008)
14. Haas, H., Brown, A.: Web services glossary, W3C Working Group Note. <http://www.w3.org/TR/ws-gloss> (Feb 2004)
15. Fensel, D., Kerrigan, M., Zaremba, M.: *Implementing Semantic Web Services: The SESA Framework*. Springer, Heidelberg (2008)
16. Van Der Aalst, W., Dumas, M., Ter Hofstede, A.: Web service composition languages: old wine in new bottles? In: *Proceedings of the 29th EUROMICRO Conference 2003, New Waves in System Architecture, Belek-Antalya* (2003)
17. Van Der Aalst, W., Ter Hofstede, A., Kiepuszewski, B., Barros, A.: Workflow patterns. *Distrib. Parallel Database* **14**, 5–51 (2003)
18. Andrews, T., Curbera, F., Dholakia, H., Golland, Y., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovic, I., Weerawarana, S.: *Business Process Execution Language for Web Services version 1.1. OASIS* (2003)
19. Fensel, D., Lausen, H., Polleres, A., De Bruijn, J., Stollberg, M., Roman, D., Domingue, J.: *Enabling Semantic Web Services: The Web Service Modeling Ontology*. Springer, Berlin (2007)
20. Roman, D., Lausen, H., Keller, U.: Web service modeling ontology (WSMO), WSMO Working Draft. <http://www.wsmo.org/TR/d2/v1.3/> (2006)
21. Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., Sycara, K.: OWL-S: semantic markup for web services, W3C Member Submission. <http://www.w3.org/Submission/OWL-S> (2004)
22. Akkiraju, R., Farrell, J., Miller, J., Nagarajan, M., Schmidt, M.T., Sheth, A., Verma, K.: Web service semantics –WSDL-S, W3C Member Submission. <http://www.w3.org/Submission/WSDL-S/> (2005)
23. Farrell, J., Lausen, H.: Semantic annotations for WSDL and XML schema, W3C Recommendation. <http://www.w3.org/TR/sawSDL/> (2007)
24. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL web ontology language semantics and abstract syntax, W3C Recommendation. <http://www.w3.org/TR/owl-semantics/> (2004)
25. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosz, B., Dean, M.: SWRL: A Semantic Web rule language combining OWL and RuleML, W3C Member Submission. <http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/> (2004)
26. Schreiber, G., Akkermans, H., Anjewierden, A., De Hoog, R., Shadbolt, N., De Velde, W.V., Wielinga, B.: *Knowledge Engineering and Management: The CommonKADS Methodology*. MIT Press, Cambridge (1999)
27. Fensel, D., Motta, E., Van Harmelen, F., Benjamins, V.R., Crubezy, M., Decker, S., Gaspari, M., Groenboom, R., Grosso, W., Musen, M., Plaza, E., Schreiber, G., Studer, R., Wielinga, B.: The unified problem-solving method development language UPML. *Knowl. Inf. Syst.* **5**, 83–131 (2003)
28. De Bruijn, J. (ed.): *The Web service modeling language (WSML)*, WSML Working Draft D16.1v0.2. <http://www.wsmo.org/TR/d16/> (2008)
29. The Object Management Group: *Meta-object facility, version 1.4*. <http://www.omg.org/technology/documents/formal/mof.htm>. (2002)
30. Gurevich, Y.: Evolving algebras: an attempt to discover semantics. In: Rozenberg, G., Salomaa, A. (eds.) *Current Trends in Theoretical Computer Science*, pp. 266–292. World Scientific, Singapore (1993)
31. Norton, B., Pedrinaci, C., Henocque, L., Kleiner, M.: 3-level behavioural models for semantic web services. *Int. Trans. Syst. Sci. Appl.* **4**, 340–355 (2008)
32. Omelayenko, B., Crubézy, M., Fensel, D., Benjamins, V.R., Wielinga, B., Motta, E., Musen, M., Ding, Y.: UPML: the language and tool support for making the semantic web alive. In: Fensel, D., Hendl, J., Lieberman, H., Wahlster, W. (eds.) *Spinning the Semantic Web*. MIT Press, Cambridge (2003)
33. Chandrasekaran, B.: Generic tasks in knowledge based reasoning: high-level building blocks for expert system design. *IEEE Expert.* **1**(3), 23–30 (1986)
34. Verma, K., Sheth, A.: Semantically annotating a web service. *IEEE Internet Comput.* **11**, 83–85 (2007)

35. Clement, L., Hately, A., Von Riegen, C., Rogers, T.: UDDI Specification Version 3.0.2. (2004)
36. Akkiraju, R., Sapkota, B.: Semantic annotations for WSDL and XML schema – usage guide, Working Group Note. <http://www.w3.org/TR/sawSDL-guide/> (2007)
37. Akhtar, W., Kopecký, J., Krennwallner, T., Polleres, A.: XSPARQL: traveling between the XML and RDF worlds – and avoiding the XSLT pilgrimage. In: *The Semantic Web: Research and Applications. Proceedings of the Fifth European Semantic Web Conference (ESWC 2008)*, Tenerife. Lecture Notes in Computer Science, vol. 5021, pp. 432–447. Springer, Heidelberg (2008)
38. Vitvar, T., Kopecký, J., Viskova, J., Fensel, D.: WSMO-lite annotations for web services. In: *Proceedings of the Fifth European Semantic Web Conference (ESWC 2008)*, Tenerife. Lecture Notes in Computer Science, vol. 5021, pp. 674–689. Springer, Heidelberg (2008)
39. Hepp, M.: Products and services ontologies: a methodology for deriving OWL ontologies from industrial categorization standards. *Int. J. Semantic Web Inf. Syst.* 2, 72–99 (2006)
40. Richardson, L., Ruby, S.: *RESTful Web Services*. O’Reilly Media, Sebastopol (2007)
41. Maleshkova, M., Kopecký, J., Pedrinaci, C.: Adapting SAWSDL for semantic annotations of RESTful services. In: *Workshop: Beyond SAWSDL at OnTheMove Federated Conferences & Workshops (OTM 2009)*, Vilamoura. Lecture Notes in Computer Science, vol. 5872, pp. 917–926. Springer, Berlin (2009)
42. Kopecký, J., Gomadam, K., Vitvar, T.: hRESTS: an HTML microformat for describing RESTful web services. In: *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2008)*, Sydney. IEEE Computer Society, Washington, DC (2008)
43. Lathem, J., Gomadam, K., Sheth, A.: SA-REST and (S)mashups: adding semantics to RESTful services. In: *Proceedings of the International Conference on Semantic Computing (ICSC 2007)*, Irvine (2007)
44. Sheth, A., Gomadam, K., Lathem, J.: SA-REST: semantically interoperable and easier-to-use services and mashups. *IEEE Internet Comput.* 11, 91–94 (2007)
45. Connolly, D. (ed.): *Gleaning Resource Descriptions from Dialects of Languages (GRDDL)*, W3C Recommendation. <http://www.w3.org/TR/grddl/> (2007)
46. Adida, B., Birbeck, M., Mccarron, S., Pemberton, S. (eds.): *RDFa in XHTML: syntax and processing*, W3C Recommendation. <http://www.w3.org/TR/rdfa-syntax/> (2008)
47. Gomadam, K., Ranabahu, A., Nagarajan, M., Sheth, A., Verma, K.: A faceted classification based approach to search and rank web APIs. In: *Proceedings of the 2008 IEEE International Conference on Web Services (ICWS 2008)*, Washington, DC (2008)
48. Elenius, D., Denker, G., Martin, D., Gilham, F., Khouri, J., Sadaati, S., Senanayake, R.: The OWL-S editor – a development tool for semantic web services. In: *Proceedings of the Second European Semantic Web Conference (ESWC 2005)*, Heraklion. Lecture Notes in Computer Science, vol. 3532, pp. 78–92. Springer, Heidelberg (2005)
49. Hess, A., Johnston, E., Kushmerick, N.: ASSAM: a tool for semi-automatically annotating semantic web services. In: *Proceedings of the Third International Semantic Web Conference (ISWC 2004)*, Hiroshima. Lecture Notes in Computer Science, vol. 3298, pp. 320–334. Springer, Berlin (2004)
50. Sabou, M.: *Building web service ontologies*. Ph.D. thesis, Vrije Universiteit Amsterdam, the Netherlands (2006)
51. Srinivasan, N., Paolucci, M., Sycara, K.: Adding OWL-S to UDDI: implementation and throughput. In: *Proceedings of First International Conference on Semantic Web Services and Web Process Composition (SWSWPC 2004)*, Hiroshima (2004)
52. Sycara, K., Paolucci, M., Ankolekar, A., Srinivasan, N.: Automated discovery, interaction and composition of semantic web services. *Web Semant. Sci. Serv. Agent World Wide Web* 1, 27–46 (2003)
53. Haarslev, V., Moller, R.: RACER: a core inference engine for the semantic web. In: *Proceedings of the Second International Semantic Web Conference (ISWC 2003)*, Sanibel Island. Lecture Notes in Computer Science, vol. 2870. Springer, Berlin (2003)
54. Jaeger, M., Rojec-Goldmann, G., Liebetrueth, C., Mühl, G., Geihls, K.: Ranked matching for service descriptions using OWL-S. In: *Kommunikation in verteilten Systemen (KiVS 2005)*, Kaiserslautern (2005)

55. Klusch, M., Fries, B., Sycara, K.: Automated semantic web service discovery with OWLS-MX. In: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate (2006)
56. Li, L., Horrocks, I.: A software framework for matchmaking based on semantic web technology. *Int. J. Elect. Commer.* **8**, 39 (2004)
57. Paolucci, M., Ankolekar, A., Srinivasan, N., Sycara, K.: The DAML-S virtual machine. In: Proceedings of Second International Semantic Web Conference (ISWC 2003), Sandial Island. Lecture Notes in Computer Science, vol. 2870, pp. 290–305. Springer, Berlin (2003)
58. HP Labs. Jena: A Semantic Web Framework for Java. <http://jena.sourceforge.net/> (2004)
59. Friedman-Hill, E.: JESS in Action. Manning, Greenwich (2003)
60. Sirin, E., Parsia, B., Hendler, J.: Filtering and selecting semantic web services with interactive composition techniques. *IEEE Intell. Syst.* **19**, 42–49 (2004)
61. Traverso, P., Pistore, M.: Automated composition of semantic web services into executable processes. In: Proceedings of the Third International Semantic Web Conference (ISWC 2004), Hiroshima. Lecture Notes in Computer Science, vol. 3298, pp. 380–394. Springer, Berlin (2004)
62. Dimitrov, M., Simov, A., Konstantinov, M., Momtchev, V.: WSMO studio – a semantic web services modelling environment for WSMO (system description). In: Proceedings of the Fourth European Semantic Web Conference (ESWC 2007), Innsbruck. Lecture Notes in Computer Science, vol. 4519, pp. 749–758. Springer, Berlin (2007)
63. Kerrigan, M., Mocan, A., Tanler, M., Fensel, D.: The web service modeling toolkit – an integrated development semantic web services. In: Proceedings of the Fourth European conference on the Semantic Web (ESWC 2007), Innsbruck. Lecture Notes in Computer Science, vol. 4519, pp. 789–798. Springer, Berlin (2007)
64. Akkermans, H., Baida, Z., Gordijn, J., Peña, N., Altuna, A., Laresgoiti, I.: Value webs: ontology-based bundling of real-world services. *IEEE Intell. Syst.* **19**, 57–66 (2004)
65. Clancey, W.J.: Heuristic classification. *Artif. Intell.* **27**, 289–350 (1985)
66. Stollberg, M.: Scalable semantic web service discovery for goal-driven service-oriented architectures. Ph.D. thesis, University Innsbruck, Austria (2008)
67. Turati, A., Valle, E.D., Cerizza, D., Facca, F.M.: Using GLUE to solve the discovery scenarios of the SWS-challenge. In: Proceedings of the Eighth Workshop on Semantic Web Services Challenge (SWSC 2007), Eindhoven, pp. 185–197 (2007)
68. Nitzsche, J., Van Lessen, T., Karastoyanova, D., Leymann, F.: BPEL for semantic web services (BPEL4SWS). In: On the Move to Meaningful Internet Systems 2007 Workshops (OTM 2007), Vilamoura. Lecture Notes in Computer Science, vol. 4805, pp. 179–188. Springer, Berlin (2007)
69. Motta, E., Domingue, J., Cabral, L., Gaspari, M.: IRS-II: a framework and infrastructure for semantic web services. In: Proceedings of the Second International Semantic Web Conference (ISWC 2003), Sanibel Island. Lecture Notes in Computer Science, vol. 2870, pp. 306–318. Springer, Berlin (2003)
70. Domingue, J., Cabral, L., Galizia, S., Tanasescu, V., Gugliotta, A., Norton, B., Pedrinaci, C.: IRS-III: a broker-based approach to semantic web services. *Web Semant. Sci. Service Agent World Wide Web* **6**, 109–132 (2008)
71. Motta, E.: Reusable Components for Knowledge Modelling: Case Studies in Parametric Design Problem Solving. IOS Press, Amsterdam (1999)
72. Hakimpour, F., Domingue, J., Motta, E., Cabral, L., Lei, Y.: Integration of OWL-S into IRS-III. In: First AKT Workshop on Semantic Web Services, KMi. The Open University, Milton Keynes (2004)
73. Motta, E., Lu, W.: A library of components for classification problem solving. IBrow (ist-1999–19005) deliverable. http://projects.kmi.open.ac.uk/ibrow/Publications/Motta_pkaw00.pdf (2001)
74. Galizia, S., Gugliotta, A., Pedrinaci, C.: A formal model for classifying trusted semantic web services. In: Proceedings of the Third Asian Semantic Web Conference (ASWC 2008), Bangkok. Lecture Notes in Computer Science, vol. 5367, pp. 540–554. Springer, Berlin (2008)
75. Pedrinaci, C., Grenon, P., Galizia, S., Gugliotta, A., Domingue, J.: A knowledge-based framework for web service adaptation to context. In: Enabling Context-Aware Web Services: Methods,

Architectures, and Technologies. Chapman & Hall/CRC Press, Boca Raton (2010)

76. Sheth, A., Gomadam, K., Ranabahu, A.: Semantics enhanced services: METEOR-S SAWSDL and SA-REST. *IEEE Data Eng. Bull.* **31**, 8–12 (2008)
77. Patil, A., Oundhakar, S., Sheth, A., Verma, K.: METEOR-S web service annotation framework. In: *Proceedings of the 13th International Conference on World Wide Web (WWW 2004)*, New York (2004)
78. Verma, K., Sivashanmugam, K., Sheth, A., Patil, A., Oundhakar, S., Miller, J.: METEOR-S WSDL: a scalable P2P infrastructure of registries for semantic publication and discovery of web services. *Int. J. Inf. Technol. Manag.* **6**, 17–39 (2005)
79. Nagarajan, M., Verma, K., Sheth, A., Miller, J., Lathem, J.: Semantic interoperability of web services – challenges and experiences. In: *Proceedings of the IEEE International Conference on Web Services (ICWS 2006)*, Salt Lake City (2006)
80. Sivashanmugam, K., Miller, J.A., Sheth, A., Verma, K.: Framework for semantic web process composition. *Int. J. Elect. Commer.* **9**, 71–106 (2005)
81. Nagarajan, M., Verma, K., Sheth, A.P., Miller, J.A.: Ontology driven data mediation in web services. *Int. J. Web Service Res.* **4**, 104–126 (2007)
82. Sivashanmugam, K., Miller, J.A., Sheth, A.P., Verma, K.: Framework for semantic web process composition. *Int. J. Elect. Commer.* **9**, 71–106 (2004–2005)
83. Oldham, N., Verma, K., Sheth, A., Hakimpour, F.: Semantic WS-agreement partner selection. In: *Proceedings of the 15th International Conference on World Wide Web (WWW 2006)*, Edinburgh (2006)
84. Verma, K., Doshi, P., Gomadam, K., Miller, J., Sheth, A.: Optimal adaptation in web processes with coordination constraints. In: *Proceedings of the International Conference on Web Services (ICWS 2006)*, Chicago. IEEE Computer Society, Washington, DC (2006)
85. Gomadam, K., Verma, K., Sheth, A., Miller, J.: Demonstrating dynamic configuration and execution of web processes. In: *Proceedings of the Third International Conference on Service-Oriented Computing (ICSOC 2005)*, Amsterdam. Lecture Notes in Computer Science, vol. 3826, pp. 502–507. Springer, Berlin (2005)
86. Gugliotta, A., Domingue, J., Cabral, L., Tanasescu, V., Galizia, S., Davies, R., Gutiérrez-Villarías, L., Rowlatt, M., Richardson, M., Stincic, S.: Deploying semantic web services-based applications in the e-government domain. *J. Data Semant.* **10**, 96–132 (2008)
87. Cardoso, J., Sheth, A. (eds.): *Semantic Web Services, Processes and Applications*. Springer, Berlin (2006)
88. Sheth, A., Verma, K., Gomadam, K.: Semantics to energize the full services spectrum. *Commun. ACM* **49**, 55–61 (2006)
89. Christensen, E., Curbera, F., Meredith, G., Weerawarana, S. (eds.): *Web Services Description Language (WSDL) 1.1*, W3C Note (2001)
90. Booth, D., Liu, C.K. (eds.): *Web Services Description Language (WSDL) version 2.0 part 0: primer*, W3C Recommendation. <http://www.w3.org/TR/wsdl20-primer/> (2007)
91. OASIS Web Services Business Process Execution Language (WSBPEL) TC: *Web services business process execution language version 2.0. Committee specification*. <http://docs.oasis-open.org/wsbpel/2.0/CS01/wsbpel-v2.0-CS01.pdf> (2007)
92. Mandell, D., McIlraith, S.: Adapting BPEL4WS for the semantic web: the bottom-up approach to web service interoperation. In: *Proceedings of the Second International Semantic Web Conference (ISWC 2003)*, Sanibel Island. Lecture Notes in Computer Science, vol. 2870. Springer, Berlin (2003)
93. Staab, S., van der Aalst, W., Benjamins, V.R., Sheth, A., Miller, J.A., Bussler, C., Maedche, A., Fensel, D., Gannon, D.: Web services: been there, done that? *IEEE Intell. Syst.* **18**, 72–85 (2003)
94. OASIS Web Services Notification TC: *Web services topics (WS-Topics) 1.3*. http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-os.pdf (2006)
95. OASIS Web Services Notification TC: *Web services brokered notification (WS-BrokeredNotification) 1.3*. http://docs.oasis-open.org/wsn/wsn-ws_brokered_notification-1.3-spec-os.pdf (2006)
96. OASIS Web Services Notification TC: *Web services base notification (WS-BaseNotification) 1.3*. http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf (2006)
97. OASIS Web Services Notification TC: *Web services notification (WSN) 1.3*. <http://www>.

- oasis-open.org/committees/tc_home.php?wg_abbrev=wsn (2006)
98. Vedamuthu, A., Orchard, D., Hirsch, E., Hondo, M., Yendluri, P., Boubez, T., Yalçinalp, U.: Web services policy 1.5 – framework, W3C Recommendation. <http://www.w3.org/TR/ws-policy/> (2007)
 99. Sheth, A.: Beyond SAWSDL: A game plan for broader adoption of semantic web services. *IEEE Intell. Syst. Trend Controv.* **22**, 8–10 (2007)
 100. Vinoski, S.: Putting the “web” into web services: interaction models, part 2. *IEEE Internet Comput.* **6**, 90–92 (2002)
 101. Papazoglou, M., Traverso, P., Dustdar, S., Leymann, F.: Service-oriented computing: state of the art and research challenges. *Computer* **40**, 38–45 (2007)
 102. Mocan A, Cimpian E.: An ontology-based data mediation framework for semantic environments. *Int. J. Semant. Web Inf. Syst.* **3**(2), 66–95 (2007)
 103. Kopecký J, Vitvar T, Bournez C, Farrell J.: SAWSDL: Semantic Annotations for WSDL and XML schema. *IEEE Internet Comput.* **11**(6), 60–67 (2007)
 104. Pedrinaci C, Domingue J.: Toward the next wave of services: linked services for the web of data. *J. Univ. Comput. Sci.* **16**(13), 1694–1719 (2010)



Glossary

AJAX (Asynchronous JavaScript and XML)

A set of technologies enabling the client-side development of interactive Web applications. In particular, it enables the exchange of data with a server, and updates to parts of a Web page without reloading the whole page.

AI (Artificial Intelligence)

An area of computer science focusing on creating machines that can engage on behaviors that humans consider intelligent. AI is the study and design of intelligent agents, which are capable of perceiving their environment and intelligently reacting to it. Artificial intelligence developments include systems which can mimic human thought, understand speech, and beat the best human chess players.

Backward chaining (or backward reasoning)

A reasoning method based on inference rules and logical implications, used in automated theorem provers and artificial intelligence applications. Backward chaining starts with a list of goals (or a hypothesis) and works backward from the consequent to the antecedent to see if there is data available that will support any of these consequents.

BPEL (Business Process Execution Language)
Closed-world assumption

An OASIS standard execution language for specifying executable business processes based on Web services.

CSS (Cascading Style Sheets)

A fundamental presumption in logic and logic reasoning stating that what is not known to be true is considered to be false. Opposite to the open-world assumption (see below).

A style-sheet language used to describe the visual appearance and format of Web sites and Web applications. CSS enables the separation of document content from document presentations, so that multiple pages can share formatting and also one document can be visualized in a multitude of different ways by simply replacing the style-sheets.

DAML (DARPA Agent Markup Language)
DAML+OIL

A markup language based on RDF aiming to support the creation of machine-readable representations for the Web. An early combination of the DAML and OIL languages (see above and below). A syntax, layered on RDF and XML, that could be used to describe sets of facts making up an ontology. DAML + OIL was a starting point for the development of OWL (see below).

Datalog	A first-order-logic query and rule language for deductive databases. Datalog is a subset of Prolog where query evaluation is carried out using bottom-up approaches.
DBpedia	A project aiming to create a dataset based on extracting linked data from articles available in Wikipedia. The resulting structured RDF data can be queried for relationships and properties associated with Wikipedia resources.
DNS (Domain Name System)	A distributed hierarchical naming system for computers, services, and resources connected in a network such as the Internet or private networks. DNS maps domain names meaningful to humans into the numerical identifiers (IP addresses) associated with networking equipment for the purpose of locating and addressing these devices globally.
DOM (Document Object Model)	An interface-oriented representation of documents in terms of nodes and a treelike structure. A DOM document can be created by a parser, or can be generated manually by users.
Dublin Core	Dublin Core usually refers to the Simple Dublin Core Metadata Element Set which has 15 metadata elements which have proven useful for describing a variety of resources. Example elements include title, creator, and date. Dublin Core Metadata Initiative (DCMI) communities are where people interested in any topic related to Dublin Core metadata can come together. Anyone who subscribes to the open mailing list can participate in a DCMI Community. There are communities for the following topics: Accessibility, Collection Description, Education, Environment, Government, Identifiers, Kernel, Knowledge Management, Libraries, Localization and Internationalization, Preservation, Registry, Scholarly Communications, Science and Metadata, Social Tagging, Standards, and Tools.
EAI (Enterprise Application Integration)	The integration of the computer applications of an enterprise so as to maximize their utility throughout the organization. The process of linking distributed applications within an enterprise in order to realize a better financial and operational competitiveness.
Endpoint (Web service endpoint)	An endpoint indicates a specific location for accessing a service using a specific protocol and data format. An association between a binding and a network address, specified by a URI that may be used to communicate with an instance of an online service.
Flickr	An image hosting and video-hosting Web site, Web services suite, and online community. A very popular Web site for

	sharing photos and currently hosts 5 billion images; the Web site is http://www.flickr.com/ .
F-Logic (Frame Logic)	F-Logic is an ontology language which is based on first-order logic, where classes and properties are modeled as terms rather than predicates. Features include, object identity, complex objects, inheritance, polymorphism, query methods, and encapsulation.
Folksonomy	A system of classification derived from the practice and method of collaboratively creating and managing tags to annotate and categorize content.
Forward chaining	A reasoning method based on inference rules and logical implications, used in expert systems, production rule systems, and artificial intelligence applications. The opposite of forward chaining is backward chaining. Forward chaining starts with the available data and uses inference rules to extract more data until a goal is reached.
FOAF (Friend Of A Friend)	A project aiming to create a Web of machine-readable pages describing people, the links between them and the things they create and do. An individual person's description is based on the FOAF ontology.
FTP (File Transfer Protocol)	FTP supports the copying of files from one host to another over the Internet (or any TCP-IP network).
GATE (General Architecture for Text Engineering)	An open source toolkit that is used for a range of natural language processing (NLP) tasks such as information extraction. The toolkit includes a desktop client for developers, a workflow-based Web application, and a Java library. GATE was originally developed at the University of Sheffield.
GRDDL (Gleaning Resource Descriptions from Dialects of Languages)	A W3C Recommendation that enables developers to extract RDF triples from an XML document.
HTML (HyperText Markup Language)	The publishing language of the Web. It is a markup language, which means that it is used to annotate a given document, in this case to describe the structure of the document (i.e., the title, headings, paragraphs, lists, quotes, and links). HTML also allows images and other objects to be embedded in the document, and can be used to create interactive forms. The last HTML specification published by W3C is the HTML 4.01 Recommendation.
HTML5	The forthcoming fifth major revision of HTML which is currently in W3C Working Draft status. HTML5 will have improvements such as native support for video playback,

	which currently depends on third-party browser plug-ins such as Adobe Flash.
HTTP (HyperText Transfer Protocol)	HTTP supports remote access to Web content over a network layer (TCP-IP – see below). HTTP functions as a request–response protocol in a client–server computing model. In HTTP, a Web browser typically acts as a client, while an application running on a computer host acts as a server.
Inference	“Inferencing” refers to the process of deriving new facts in a knowledge base on the basis of two sources: (a) other facts that have already been represented in the knowledge base, and (b) inference rules that are specified as part of the ontology underpinning the knowledge base.
Information extraction	A natural language processing (NLP) task that aims to obtain structured information from unstructured text.
Information retrieval	The science of searching for documents and searching for information within documents that match a given user query.
IP (Internet Protocol) address	A number that is assigned to any device connected to an IP network. The Internet Protocol is used to route data packets between networks and IP addresses are used to specify the locations of the source and destination nodes in the respective networks.
IRI (Internationalized Resource Identifier)	A generalization of the Uniform Resource Identifier (URI), which in turn is a generalization Uniform Resource Locator (URL). Unlike URIs, which are limited to the English language-only ASCII character set, IRIs may contain characters from the Universal Character Set (also known as Unicode), which covers many of the world’s languages.
JSON (JavaScript Object Notation)	A lightweight, text-based data-interchange format. It is primarily used to transmit data between a server and Web application, serving as an alternative to XML. Note that, despite its origins as a derivative of the JavaScript programming language, JSON is language independent.
Knowledge Acquisition	The process of obtaining knowledge from a subject-matter expert, which is then used in developing an expert or knowledge-based system. This knowledge can be represented as a set of IF-THEN style rules or in some other common knowledge representation format.
Knowledge base	A database of the knowledge (e.g., basic facts and IF-THEN rules) of a particular subject domain that forms part of an expert or knowledge-based system.

Knowledge engineering	The process of building an expert or knowledge-based system. More narrowly, it can refer to the process of translating the knowledge of a subject-matter expert into the knowledge base of the expert or knowledge-based system.
Knowledge representation	The technique of formally coding knowledge in a knowledge base. Knowledge representation can also refer to formally coded knowledge.
LarKC (Large Knowledge Collider)	A platform for massive distributed incomplete reasoning that aims to remove the scalability barriers of current existing reasoning systems for the Semantic Web. It is being developed within an EU FP7 project of the same name (http://www.larkc.eu/).
Linked Open Data (also "Linked Data")	A recommended best practice for exposing, sharing, and connecting pieces of data, information, and knowledge on the Semantic Web using URIs and RDF (see http://linkeddata.org/).
Logic	Logic is formally a branch of mathematics, which explores the expressive power of formal systems and the deductive power of formal proof systems. In computer science, research focuses on logic systems that are computationally feasible.
Mashup	A graphical lightweight process (composite service) description, described as an aggregation of individual graphical services, including both WSDL (see below) and RESTful (see below), connected through a simple data flow which implicitly offers a basic workflow as well.
Materialisation	Total materialization involves computing all entailed statements at load time. While this introduces additional reasoning cost when loading statements into a repository, the desirable consequence is that query evaluation can proceed extremely quickly.
Meta Content Framework (MCF)	MCF is a specification of a format for structuring metadata about Web sites and other data, developed by Ramanathan V. Guha between 1995 and 1997.
MetaData	MetaData is structured data about data. Metadata is used to facilitate the machine understanding, use, and management of data.
Microformats	Microformats are a set of simple, open data formats built upon existing and widely adopted standards. This approach allows software to process information intended for end-users such as contact information (hCard), geographic coordinates (geo), and calendar events (hCalendar) automatically. Microformats take advantage of the class and rel

	attributes of (X)HTML to embed metadata in a machine consumable way.
Natural Language Processing (NLP)	Natural Language Processing is a range of computational techniques for analyzing and representing naturally occurring text (free text) at one or more levels of linguistic analysis (e.g., morphological, syntactic, semantic, and pragmatic) for the purpose of achieving humanlike language processing for knowledge-intensive applications.
NeOn (NETworked ONtologies)	NeOn is a project aimed to advance the state of the art in using ontologies for large-scale semantic applications in distributed organizations, particularly, improving the capability to handle multiple networked ontologies that exist in a particular context, are created collaboratively, and might be highly dynamic and constantly evolving. NeOn provides methodological and tool support for developing and managing a new generation of semantic applications: the NeOn toolkit, which is an open source multi-platform ontology engineering environment providing comprehensive support for the ontology engineering life cycle. See http://www.neon-project.org/ for more details.
OASIS (Organization for the Advancement of Structured Information Standards)	OASIS is a not-for-profit consortium that drives the development, convergence, and adoption of open standards for the global information society. The consortium has produced Web services standards such as WS-BPEL (Web Services Business Process Execution Language) along with standards for security and e-business XDI (XRI Data Interchange) and ebXML (Electronic Business using eXtensible Markup Language) respectively.
OIL (Ontology Inference Layer or Ontology Interchange Language)	OIL was an ontology language based on concepts developed in Description Logic (DL) and frame-based systems and was compatible with RDFS. Much of the work in OIL was subsequently incorporated into DAML+OIL (see above) and the Web Ontology Language (OWL) (see below).
Ontology	In the artificial intelligence community, the most agreed definition of ontology is due to Gruber who defines an ontology as: “a formal, explicit specification of a shared conceptualization.” A functional definition that defines ontologies by what they are for, rather than what they are: “An ontology defines (specifies) the concepts, relationships, and other distinctions that are relevant for modelling a domain.”
Ontology population	Ontology population is the process of adding new instances of concepts/relations into an ontology, usually based on information related to terms and synonyms.

OpenID	An open, decentralized standard for authenticating users which can be used for access control, allowing users to log on to different services with the same digital identity where these services trust the authentication body.
Open Graph	A Facebook technology, which enables any Web page to become a rich object in a social graph. For instance, this is used on Facebook to enable any Web page to have the same functionality as a Facebook Page. Open Graph is based on RDFa.
Open-world assumption	Open-world assumption is the assumption that the truth-value of a statement is independent of whether or not it is known by any single observer or agent to be true. It is the opposite of the closed-world assumption (see above), which holds that any statement that is not known to be true is false.
OWL (Web Ontology Language)	A semantic markup language for publishing and sharing ontologies on the World Wide Web. OWL incorporates three variants: Lite, DL, and Full with a growing level of expressiveness. OWL became a formal W3C recommendation in February 2004.
OWL2 (Web Ontology Language 2)	A new version of OWL based on the experiences in using OWL. OWL2 introduces a number of new profiles: OWL2 EL is a fragment that has polynomial time reasoning complexity; OWL2 QL is designed to enable easier access and query to data stored in databases; and OWL2 RL is a rule subset of OWL2. OWL2 has been a W3C recommendation since October 2009.
OWL-S (Ontology Web Language for Services)	OWL-S aims to provide building blocks for encoding rich semantic service descriptions that builds naturally upon OWL. The OWL-S approach consists of an upper ontology for services with three interrelated sub-ontologies. Firstly, the profile is an ontology for describing the service functionalities in order to advertise the service and match it with the requests. Secondly, the process model is an ontology supporting behavioral descriptions incorporating service invocation, enactment, composition, monitoring, and recovery. Lastly, the grounding ontology bonds the process model with detailed specifications of the service encoded in WSDL (see below).
N3 (Notation 3)	N3 is a shorthand non-XML serialization of RDF models, designed with human-readability in mind. Moreover, N3 is far more compact and readable than XML RDF notation.
Peer to Peer (P2P)	A flat network hierarchy in which clients interact directly without the intervention of mediating servers.

Powerset	A Microsoft-owned company that is developing a natural language search engine for the Internet.
Prolog	A general purpose logic programming language associated with artificial intelligence and computational linguistics.
Protégé	Protégé is a free, open source ontology editor and knowledge base framework. The Protégé platform supports two main ways of modeling ontologies: via the Protégé-Frames and Protégé-OWL editors. Within Protégé-Frames, an ontology consists of a set of classes organized in a subsumption hierarchy to represent a domain's salient concepts, a set of slots associated to classes to describe their properties and relationships, and a set of instances of those classes. Protégé-OWL is an extension of Protégé that supports OWL where an OWL ontology may include descriptions of classes, properties and their instances. Given such an ontology, the OWL formal semantics specifies how to derive its logical consequences, that is, facts not literally present in the ontology, but entailed by the semantics.
R2RML (RDB2RDF mapping language)	A language for mapping relational data and relational database schemas into RDF and OWL.
Racer	A core inference engine for the Semantic Web. In particular, Racer is a description logics inference engine.
RDF (Resource Description Framework)	A general-purpose language for representing information in the Web. The RDF data model consists of a set of statements, each containing a subject, a predicate, and an object.
RDFa (RDF in attributes)	A W3C Recommendation that adds a set of attribute level extensions to XHTML for embedding RDF triples within Web documents.
RDQL (RDF Data Query Language)	A query language for extracting information from RDF graphs. RDQL provides a way of specifying a graph pattern that is matched against the graph to yield a set of matches.
Reasoning	The process of drawing inferences and conclusions from available information or data. These inferences can be: (a) Deductive determining a conclusion, for example, using a rule and its precondition to infer a conclusion. (b) Inductive determining a rule, that is, learning a rule after numerous examples of conclusions following a specific precondition. (c) Abductive determining the precondition. It is using the conclusion and the rule to support that the precondition could explain the conclusion.
Reification	Reification is the ability in RDF to treat a statement as a Resource, and hence to make assertions about that statement.

Repository (RDF Repository)	A purpose-built database for the storage and retrieval of RDF triples. Unlike a relational database, an RDF Repository is optimized for the storage and retrieval of triples (subject, relation, and object).
REST (Representational State Transfer)	REST is a style of software architecture for distributed hypermedia systems such as the World Wide Web. The term Representational State Transfer was introduced and defined in 2000 by Roy Fielding in his doctoral dissertation.
RIF (Rule Interchange Format)	A W3C standard that was developed to facilitate the sharing and reuse of rulesets. RIF comprises a set of interconnected dialects representing rule languages with various features. RIF became a W3C Recommendation in June 2010.
RSS feed (Really Simple Syndication or Rich Site Summary feed)	A simple format used to publish frequently updated content such as blog entries, news headlines, audio, and video, that a user can subscribe to using an RSS aggregator.
SameAs (OWL:SameAs)	A built-in OWL property that links an individual to an individual indicating that two URI references actually refer to the same thing; the individuals have the same “identity.” This property is used to link datasets that form Linked Open Data.
SAWSDL (Semantic Annotations for WSDL and XML Schema)	SAWSDL defines how to add semantic annotations to various parts of a WSDL document such as input and output message structures, interfaces, and operations.
Schema	Schema is used to define the structure for data. XML Schemas define the structure of XML documents through languages such as the Document Type Definition (DTD) or XML Schema languages. Database schemas define the structure of the data contained in a database. For a relational database, the schema definition will include a specification of a database’s table, fields, and relationships.
SearchMonkey	A Yahoo! service, which allows developers and site owners to use structured data to make Yahoo! Search results more useful and visually appealing, and to drive more relevant traffic to their sites.
Semantic Annotation	A piece of semantic metadata added to a document; for example, in WSDL semantic annotations contain semantic information about Web services.
Semantic Web Layer Cake	A diagram that shows the technologies of the Semantic Web, represented as blocks layered as one technology builds on another; the layer cake shows not only the existing technologies but also a roadmap for more advanced technologies, especially leading toward technologies for trust; see http://en.wikipedia.org/wiki/Semantic_Web_Stack .

Serialization	A transfer format for an abstract data model such as RDF, intended for interoperable communication in a computer network; for example, the RDF data model can be serialized in RDF/XML or in Turtle.
SESAME	An open-source framework enabling the storage, inferencing, and querying of RDF data in the programming language Java; currently hosted at http://www.openrdf.org/ .
Sindice	An infrastructure to process, consolidate, and query the Web of Data, which collects data and metadata especially from RDF, RDFa, and Microformat documents and allows searching by text or other metadata; currently located at http://sindice.com/ .
SIOC (Semantically-Interlinked Online Communities)	An ontology for data from online communities (e.g., message boards, wikis, and weblogs), commonly used in conjunction with FOAF (see above); submitted to the W3C at http://www.w3.org/Submission/sioc-spec/ and hosted at http://sioc-project.org/ .
SKOS (Simple Knowledge Organization System)	A common data model for sharing and linking knowledge organization systems (such as thesauri, taxonomies, classification schemes, and subject heading systems) via the Web; defined at http://www.w3.org/TR/skos-reference/ .
SNOMED (Systematized Nomenclature of Medicine)	A hierarchical classification system and a collection of medical terminology covering most areas of clinical information such as diseases, findings, procedures, microorganisms, pharmaceuticals, etc.; owned by the International Health Terminology Standards Development Organisation at http://www.ihtsdo.org/ .
SOA (Service-Oriented Architecture)	A loosely defined IT architecture that focuses on decomposing systems into Web services (see below) usually according to a set of business requirements.
SPARQL	A query language for RDF data that supports querying diverse data sources, with results in the form of a variable-binding table, or an RDF graph; also a protocol built on top of HTTP that enables the sending of queries to external servers; defined at http://www.w3.org/TR/rdf-sparql-query/ .
SWOOGLE	A Semantic Web search engine (a portmanteau of Semantic Web and Google) that allows textual and metadata queries to find ontologies or Semantic Web documents; currently located at http://swoogle.umbc.edu/ .
SWOOP	A tool for creating, editing, and debugging OWL ontologies that employs a Web-browser metaphor for its design and usage; currently available via http://www.mindswap.org/2004/SWOOP/ .

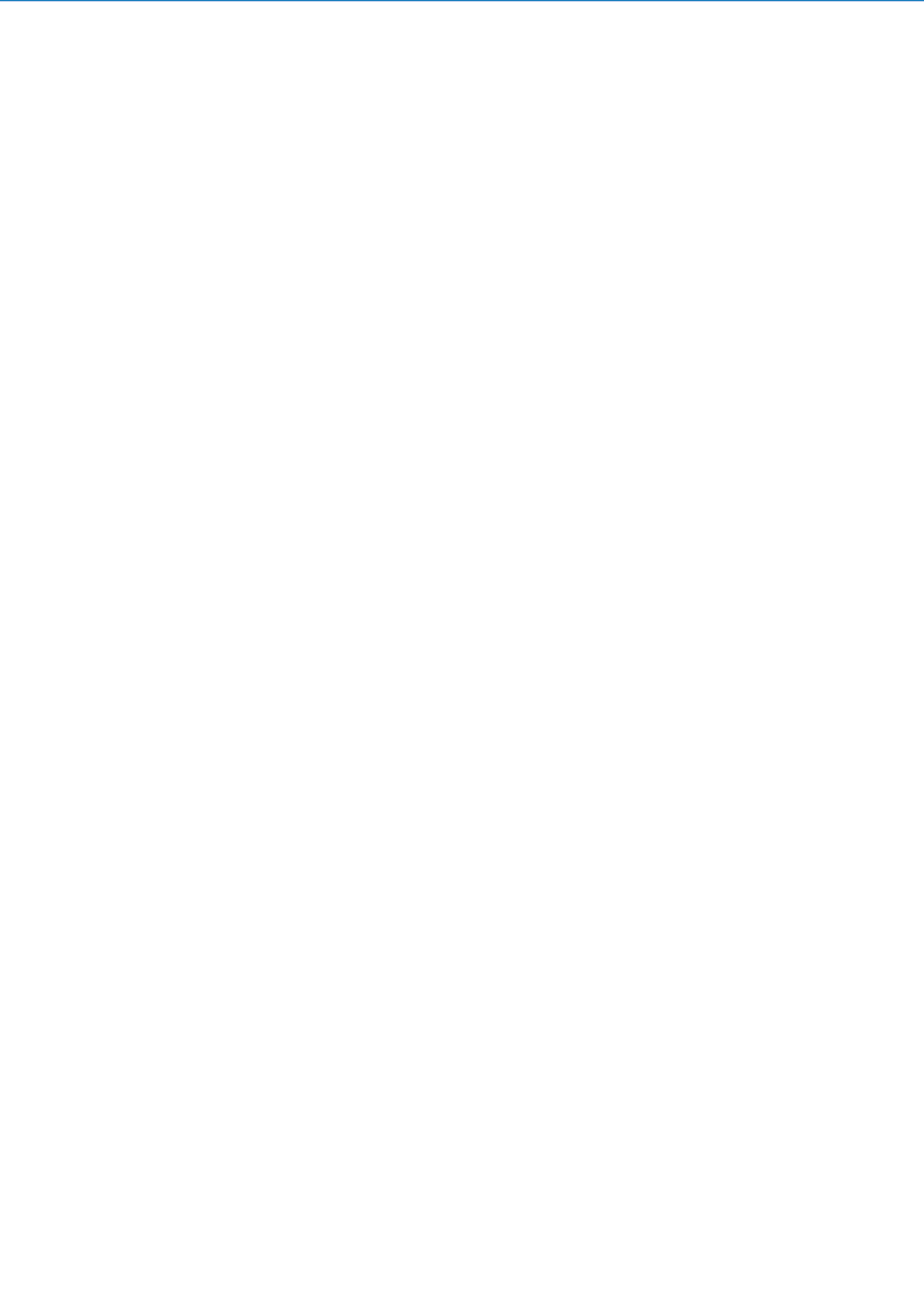
SWRL (Semantic Web Rule Language)	A rule language for the Semantic Web, submitted to the W3C. Most SWRL rules can now be exchanged via RIF (see above). SWRL is defined at http://www.w3.org/Submission/SWRL/ .
TCP-IP	A set of network protocols (especially the Internet Protocol IP and the Transmission Control Protocol TCP) that drive the Internet. For instance, HTTP (see above) builds upon TCP. TCP is defined at http://tools.ietf.org/html/rfc793 and IP is defined at http://tools.ietf.org/html/rfc791 .
TopBraid Composer	A commercial Semantic Web modeling and application development environment available from http://www.topquadrant.com/products/TB_Composer.html
Triple	In RDF, the ordered 3-tuple of <subject, predicate, object>, where a subject is a URI or a blank node; a predicate is a URI, and an object is a URI, a blank node or a literal; and a blank node is an unnamed resource, commonly used to represent a structured value that need not be individually addressable. An example use of a blank node would be a postal address (with separate properties for street, city, country, and so on) attached as the mailing address of a person or an organization.
Triple Store	A software system for the storage and retrieval of RDF data, often in “named graphs” – sets of RDF triples collectively identified with a single URI. Retrieval is often carried using the SPARQL and protocol (see above). Triple stores additionally often provide inference capabilities.
Turtle	A serialization syntax for RDF that forms a subset of the N3 syntax (see above). Turtle is defined at http://www.w3.org/TeamSubmission/turtle/ .
Twitter	A microblogging system (every “tweet” is up to 140 characters) with optimized search through the mentions of users (who write about me or to me) and by mentions of topics or tags. Twitter is currently located at http://twitter.com/ .
UDDI (Universal Description Discovery and Integration)	A business-oriented registry of Web services, tying business entities to the Web services they provide. There was also a public UDDI registry but this is no longer in use. Currently defined at http://uddi.xml.org/ .
UML (Unified Modeling Language)	A standard graphical modeling language for software engineering, including the modeling of use cases, components, activities, processes, and data schemas. UML is currently defined at http://www.uml.org/ .
URI (Uniform Resource Identifier)	A string with a defined format that is used to identify a resource. A resource may be on a Web (e.g., a Web page,

URL (Uniform Resource Locator)	<p>an image, or a data source) or it may be something more abstract (e.g., a person or an e-mail address). URIs are current defined by http://tools.ietf.org/html/rfc3986.</p> <p>A URL is a URI that serves the purpose of both identifying a resource and also describing its network location so that it can be found and accessed. URLs are defined together with URIs (see above).</p>
URN (Uniform Resource Name)	<p>A URN is a URI that serves predominantly as a stable resource name, without direct ties to a location of the resource. URNs are defined together with URIs (see above).</p>
WATSON	<p>A gateway for the Semantic Web that crawls and analyzes semantic content on the Web and provides efficient query and keyword search. WATSON is currently located at http://watson.kmi.open.ac.uk/.</p>
W3C (World Wide Web Consortium)	<p>A standardization consortium formed by industrial and research organizations to create technology standards for the Web. Among the standards that W3C has produced are HTML, XML, RDF, OWL, and WSDL.</p>
Web 2.0	<p>Web 2.0 introduces no real technologies over the Web but emphasizes the notion of prosumers. Prosumers both produce and consume content. Prime examples of Web 2.0 sites include Wikipedia, YouTube (see below), and social networking sites such as Facebook.</p>
Web APIs	<p>A Web Service (see below) implemented using native Web technologies (especially HTTP, JSON, and XML (all also in this glossary)) that gives programmatic access to the functionalities of a Web site, in addition to the human access through the HTML pages.</p>
Web of Data	<p>A term that groups the data sources on the World Wide Web, available in machine-processible formats (see XML, JSON, RDF) rather than in a human-oriented form such as presented in HTML. The Web of data is directly usable by programs that can download and process the data.</p>
Web Service	<p>A software system that makes a piece of business functionality available through standardized computer networks and protocols. The interface of a Web Service is often formally described using WSDL (see below), facilitating the creation of client programs. The definition of a Web Service can be found at http://www.w3.org/TR/ws-arch.</p>
Wiki	<p>A Web authoring system optimized for the simple authoring of Web content and hyperlinking between entries in a single system, with a special syntax that is simpler than HTML. A Wiki will often have features for collaborative authoring,</p>

Wikipedia	<p>such as a page history and change notifications. Wikipedia (below) is the most famous example of a Wiki.</p> <p>A free, multi-language, hyperlinked online encyclopedia created and managed by volunteers. By default, everybody is allowed to edit Wikipedia entries (in comparison to traditional encyclopedias edited by small teams of experts), resulting in a wide breadth of topics and coverage. Wikipedia is currently located at http://wikipedia.org/.</p>
Wordnet	<p>A lexical database for the English language, containing basic information about English words, including synonyms and is-a relationships (hyponyms). Wordnet is often used in conjunction with ontologies to provide breadth. It is currently hosted at http://wordnet.princeton.edu/.</p>
WSDL (Web Services Description Language)	<p>An XML language for describing the interfaces and end-points of Web services. Currently defined at http://www.w3.org/TR/wsdl20.</p>
WSMO (Web Service Modeling Ontology)	<p>An ontology supporting the semantic description of Web services. Among the top-level concepts of WSMO in addition to services are goals, to capture the client perspective, and mediators, to resolve heterogeneities. WSMO is currently defined at http://wsmo.org/TR/d2.</p>
XML (Extensible Markup Language)	<p>A standard representation for structured and semi-structured data. XML is widely deployed supporting data exchange and storage. XML is currently defined at http://www.w3.org/TR/xml.</p>
XSLT (XSL Transformations)	<p>A functional language supporting transformations between XML languages. XSLT is closely tied with XPath (a language for addressing and retrieving nodes from XML documents) and is currently defined at http://www.w3.org/TR/xslt20.</p>
YouTube	<p>A video-hosting, sharing, and discussion Web site launched in 2005 and reported in May 2010 to serve over two billion video views daily. It is now owned by Google and is located at http://youtube.com/.</p>

Acknowledgments

We would like to thank the following for their support in creating this glossary: Neil Benn, Jacek Kopecky, Bassem Makni, and Maria Maleshkova.



Index

A

Analysis, 913–915, 917, 922, 929–935, 962–963, 967, 968
Annotations, 117–154, 913–917, 919–929, 931, 934, 942, 952, 955, 960, 962–963, 966, 968, 969
Applications, 585–605, 608–614, 616
Approximation, 455

B

Bio2RDF, 201, 209, 333–339, 357, 456, 723, 730
Blogs, 469, 471–473, 475–478, 482–490
British Broadcasting Corporation (BBC), 25, 34, 200, 219, 552
Business, 621–657

- document ontology, 816, 820
- model, 792, 794–804, 813–816, 834
- process model, 792, 796, 797, 799, 801, 804–816, 834–836
- semantics, 797, 804, 813

Business-to-business (B2B) electronic commerce, 796–822, 841

C

Closed-world assumption, 14, 21
Composition, 978–981, 983, 984, 988, 996, 1000, 1002–1004, 1006, 1011, 1012, 1014–1017, 1026–1029
Core components, 816, 818–820, 836
Cultural heritage, 913, 914, 945, 949–962, 964–969

D

Data integration, 197, 207, 212, 216, 217, 219, 222, 706–708, 719, 721, 722, 724, 726, 729–730
Data management, 219
Datasets, 60, 234, 235, 242–246, 251, 253, 259–271, 273, 277–286, 288, 291, 292, 307, 319, 320, 330, 338–341, 349, 354, 357, 358
DBpedia, 45, 48, 55, 60, 64, 65, 67, 92, 95, 148, 180, 186, 187, 200–202, 204, 206, 209, 213, 217–219, 245, 252, 253, 268, 269, 284, 286, 303–306, 308–316, 319–326, 339, 340, 342, 343, 355, 357, 456, 476
Description logic, 14–16, 19–21, 366–378, 383–388, 393–394

Discovery, 979, 980, 982, 984, 988, 990, 992, 996, 1003–1006, 1010, 1012–1014, 1016, 1018, 1026, 1028
Distribution, 444, 447, 448, 451, 454, 461

E

EBUCore, 916, 936–937, 942
eGovernment, 849–904
Endpoint, 302, 307, 326, 334, 338, 354, 359
Enterprises, 626–627, 631, 637, 638, 640, 641, 643
Entity disambiguation, 92, 95–96
e-Science, 701–732
Evaluation of semantic annotation, 77–113
eXtensible HyperText Markup Language (XHTML), 159–161, 164–172, 174, 176, 178, 181, 183, 186–189, 818

F

Facebook, 32–33
Folksonomy, 766–768, 770
Friend-of-a-Friend (FOAF), 23, 168–178, 180, 183, 474, 708, 771, 832, 891
Full-cycle benchmarking, 258, 260–261, 273, 277, 278
Future, 581–616

G

Gleaning Resource Descriptions from Dialects of Languages (GRDDL), 160–162, 166–168, 178, 188
Google, 8, 9, 26, 33–34, 37

H

High-performance computing, 447–448, 458
Hypertext, 157–189
HyperText Markup Language (HTML), 8, 9, 10, 12, 15, 17, 24, 26, 29, 159–167, 179, 181, 185, 188–189, 217, 708, 818
HyperText Transfer Protocol (HTTP), 54, 193, 195, 198, 203–206, 208, 212, 221, 225, 709

I

Information integration, 741, 746–751, 773–775, 780

K

Knowledge management, 737–782
Knowledge representation, 11, 367

Knowledge sharing, 486, 495, 501, 740, 741, 745–746, 756–757, 765, 767, 773, 778
 Knowledge transfer, 732

L

Large applications, 301
 Linked data, 193–195, 197–198, 200–203, 205, 207–209, 211–214, 216–225, 862, 864, 866, 871, 877, 882–890, 892–899, 904
 Linked open data (LOD), 126, 151, 756–757, 777, 779
 Loading performance, 250, 258–261, 265, 266, 277–278, 280
 Logic, 13–22, 24, 25, 38, 368, 369, 372

M

Market needs, 641, 642
 Matchmaking, 826, 830–831, 833
 Mediation, 982, 983, 988, 989, 991, 992, 996, 1003, 1004, 1006–1007, 1009, 1012–1014, 1022, 1025–1026, 1028–1029
 Metadata, 119–121, 130, 131, 137–140, 147, 152, 154
 Microblogging, 475, 482, 486–487, 502
 Microformats, 17, 32, 34, 157–189, 435, 476, 479, 484, 499, 646, 697, 698, 765, 996, 999, 1029, 1032
 MPEG-7, 916–922, 927, 928, 931, 941, 962–963
 Multimedia schema, 914
 Multimedia search, 914
 Multi-paradigm search, 80, 93, 100, 109, 111
 Museums, 148, 149, 303–305, 558, 589, 827, 829, 922, 953, 955, 961, 965, 966

N

Natural language processing (NLP), 24, 597, 645, 663, 775

O

Ontologies, 5, 15, 16, 18, 20, 22–24, 26, 30, 34, 45, 46, 50–52, 58, 59, 61–62, 64–65, 67, 69–71, 366, 369–375, 380, 381, 383–394, 507–564
 – engineering, 518, 527–532, 537, 538, 561, 562
 – learning, 527, 532–538, 561, 562
 – management, 538, 561
 – mapping, 771, 774, 775, 782
 Ontology-based faceted browsing, 97–100
 Ontology-based information extraction, 102, 112
 Ontology-based information systems, 538, 549
 Open Graph (Protocol), 32–33
 Open innovation, 626
 Open-world assumption, 21

Orchestration, 983, 990, 1002–1004, 1006, 1008, 1009, 1011–1012, 1019–1021, 1024, 1025, 1029
 OWL. *See* Web ontology language
 OWL2, 16, 20, 31
 OWL-S, 984–988, 994, 1000–1003, 1006, 1008–1009, 1012, 1021, 1027–1028

P

Pattern
 – graph pattern, 302, 308, 309, 314–316, 318, 319, 329, 331, 342, 347, 351, 356, 357
 – triple pattern, 301, 302, 307, 308, 319–321, 325, 338, 347, 355, 357
 Predictions, 583, 585, 603, 605, 607–612
 Provenance, 706–708, 710, 713, 727, 728, 732
 PSI. *See* Public sector information
 Public administration, 867, 870, 871
 Public sector information (PSI), 853, 863, 871, 876, 877, 885, 886, 888, 889, 892–894

Q

Quality, 703, 706–708, 710, 713, 716–717, 725, 732
 Query
 – ASK query, 302, 307, 323, 325, 347, 359
 – CONSTRUCT query, 323–326, 336, 347
 – DESCRIBE query, 302, 307, 323–327, 347
 – evaluation, 236, 237, 239, 241, 248–251, 253, 258–261, 264, 266–273, 276, 278–283
 – query language, 301, 302, 308, 326, 327, 346–351, 354, 356, 359, 360
 – SELECT query, 302, 307, 308, 311–313, 315–323, 326, 331–333, 337, 340, 341, 343, 346–351, 353, 356, 358, 359

R

RDF. *See* Resource Description Framework
 RDFa, 17, 32, 34, 56, 120, 147–149, 157–189, 197, 207, 208, 215, 216, 435, 481–482, 484, 492, 496, 498–499, 623, 687, 697, 698, 727, 765, 864, 999, 1029, 1032
 RDF database, 150, 236, 238, 239, 258, 270–272, 275, 276, 287, 293
 RDF schema (RDFS), 15–17, 19–22, 31, 51, 120, 121, 124, 132–140, 144, 145, 147, 149–154, 176, 187, 422
 Reification, 85, 129–132, 152, 382, 414, 520
 Representational State Transfer (REST), 55, 195, 706, 979, 996, 997, 999–1000, 1029, 1031
 Resource Description Framework (RDF), 15–22, 25, 27, 31, 34, 50–65, 67, 69–71, 80–83, 86, 88, 97, 98, 100, 103, 106–107, 112–113, 117–154, 159–162,

165–169, 171–180, 182–183, 185–189, 193, 197, 198, 200–210, 212–219, 221, 223, 225, 233, 235–249, 253, 254, 257–261, 267–277, 279–280, 284–287, 289–294, 301–317, 319–327, 329–334, 336, 338–339, 342–347, 350–359, 366, 370–375, 379–380, 382–385, 387, 388, 393, 400, 401, 403–405, 407, 409, 410, 417, 419, 422–427, 432, 433, 435, 437, 442–443, 451–455, 458, 460, 461, 463, 471, 477–478, 480–485, 487, 491–495, 497–499, 503, 519, 522, 526, 540, 547, 550–552, 555, 558, 561–563, 585, 589, 600–601, 623, 628, 642, 644, 661, 663–672, 674–675, 684, 689–692, 698, 708–710, 713–715, 717, 721–728, 730, 733, 750, 756, 764, 771, 778–779, 819, 851, 858, 861, 863–865, 871, 875, 877–879, 881–882, 885, 889–896, 916, 918–924, 937, 938, 940, 941, 944, 947, 949, 957, 958, 960, 962, 966, 984, 986, 987, 999, 1029, 1032

RIF basic logic dialect (RIF-BLD), 400–421, 425, 426, 429, 432–433, 436

RIF Framework for Logic Dialects (RIF-FLD), 17, 399–404, 406, 407, 411, 414, 419, 420, 429, 433, 435, 436

Roadmap, 234, 401, 590, 607–608, 611–614, 633, 645, 889, 935

Rule interchange, 422

Rule Interchange Format (RIF), 16–18, 20–22, 31

Rule systems, 400, 402–404, 419

S

SAWSDL. *See* Semantic annotations for WSDL

Scalability, 443, 453–456, 461–463

Selection, 451, 454–455

Semantic annotation, 77–113

Semantic annotations for WSDL (SAWSDL), 18, 526, 554, 978, 984, 991–995, 997, 999, 1000, 1013, 1028, 1029, 1031, 1032

Semantic desktop, 764, 770–773, 781

Semantic multimedia, 914–923, 929–935, 945, 962–966, 968

Semantic offering, 633–638

Semantic publishing, 17, 725–727, 729

Semantic repository, 231–294

Semantic repository engines, 235–240, 270–276, 280

Semantic repository performance validation, 233–235, 254, 258–261, 263, 267, 281, 290

Semantic search, 744, 759–765, 776

Semantic web (SW), 120–122, 136–139, 147–154, 194–195, 197, 206, 208, 217, 219–222, 225,

400–437, 441–463, 728, 851–854, 864, 866, 872–874, 886, 889, 898, 901

- applications, 50, 51, 53, 538, 558–560, 672, 673, 678–696
- architecture, 43–71
- landscape, 689
- languages and standards, 47–51, 55, 61, 71
- search, 659–698
- search engines, 659–698

Service Interface for Real Time Information (SIRI), 34–36

Service-Oriented Architecture (SOA), 979, 981, 1027

Simple Knowledge Organization System (SKOS), 16, 21–22, 32, 303, 481, 520

Simple Protocol and RDF Query Language (SPARQL), 17, 20, 31, 32, 299–360

SIRI. *See* Service Interface for Real Time Information

SOA. *See* Service-oriented architecture

Social networking, 37, 38, 469–478, 481, 482, 485, 486, 493, 498–500, 595, 639, 714, 715, 764, 823

Social web, 25, 468–482, 486, 489, 493, 497–501, 526, 592, 611, 863, 945, 947

SPARQL. *See* Simple Protocol and RDF Query Language

SW. *See* Semantic web

Swoogle and Watson, 661, 667–696, 698

T

Technologies, 583–616

Technology transfer, 645

Television, 913, 936–937, 945, 966, 967

Text analytics, 775–776

Text mining, 775–776

Tourism ontologies, 827–828

Trends, 581–616

Trip planning applications, 826–833

Trust, 706–708, 710, 720, 727, 732

TV-Anytime, 916, 918–919, 937–941, 943, 949, 964

U

UN/CEFACT's modeling methodology (UMM), 808–813, 815, 835, 836

Uniform Resource Identifiers (URIs), 50, 193, 195–197, 200, 201, 203–207, 210, 217–218, 221, 225, 244, 683, 709

Uniform Resource Locators (URLs), 8, 9, 203

Usability, 731

V

Vision, 583, 585, 586, 590, 593, 594, 598, 602–611, 616

W

Web 2.0, 22, 78, 144, 159, 179, 194, 201, 339, 468–470, 472, 480, 490, 498, 501, 766, 767, 769, 773, 781

Web of data, 17, 24, 25, 33, 49, 54, 55, 71, 95, 111, 113, 121, 139, 154, 179, 191–225, 709, 721–723, 725, 728, 730, 732, 733, 757, 777–779, 842, 1032

Web Ontology Language (OWL), 5, 15–22, 31, 32, 51, 205, 369–398, 409, 412, 414, 421, 426–431, 438, 442, 443, 453, 456, 458, 461, 490, 492, 498, 520, 628, 708, 836, 980

Web Service Modeling Ontology (WSMO), 526, 838, 978, 982, 984, 988–991, 994–1000, 1003–1009, 1012, 1019–1022, 1024–1026, 1028, 1029, 1032

Web services, 977–1032

Wikis, 469, 471–473, 482, 486, 492

X

XHTML. *See* eXtensible HyperText Markup Language