Semantic e-Science

Volume 11

Editors
Huajun Chen
Yimin Wang
Kei-Hoi Cheung

Annals of Information Systems

Springer

Annals of Information Systems

Volume 11

For further volumes:
http://www.springer.com/series/7573

Huajun Chen · Yimin Wang · Kei-Hoi Cheung
Editors

Semantic e-Science

Springer

Editors
Huajun Chen
College of Computer Science
Zhejiang University
Zheda Road 38
310058 Hangzhou
China, People's Republic
huajunsir@zju.edu.cn

Kei-Hoi Cheung
Center for Medical Informatics
Yale University School of Medicine
Cedar St. 333
06520-8009 New Haven
CT, USA
kei.cheung@yale.edu

Yimin Wang
Lilly Singapore Centre for
Drug Discovery Pte Ltd.
Biomedical Grove 8A
138648 Singapore
#02-05 Immunos
Singapore
wangyimin@lilly.com

ISBN 978-1-4419-5902-7 e-ISBN 978-1-4419-5908-9
DOI 10.1007/978-1-4419-5908-9
Springer New York Dordrecht Heidelberg London

Library of Congress Control Number: 2010930503

Preface

The advancement of modern science is growing more and more dependent on the Internet technology. Scientists rely upon it to retrieve valuable information and knowledge, communicate research ideas, have virtual meetings, exchange resources, and coordinate collaboration. As a matter of fact, the Internet has become an indispensable media for various scientific research endeavors in many kinds of disciplines. Specifically, the rapid increase in complexity and scope of modern science research brings up a protruding challenge, which is to support world-wide scale collaboration and coordinated resource sharing across disciplinary, organizational, social, and cultural boundaries. New discovery normally resides in those boundaries. Researchers working on one aspect typically are required to look into those boundaries, utilize resources from other research institutes, and explore knowledge from other fields from perspectives of a completely different scientific disciplinary. A typical case study is integrative neuroscience research. For example, the study of neurodegenerative diseases such as Parkinson, Alzheimer, Huntington needs to combine knowledge and resources from a good many research institutions. The activities normally span a wide range of the disciplines such as psychiatry, neurology, microscopic anatomy, neuronal physiology, biochemistry, genetics, molecular biology, and bioinformatics. These types of requirements require a new generation of information infrastructure to enable scientific investigation performed through distributed global collaborations between scientists and their resources. The term e-Science was thus brought up to refer to such kind of computing infrastructure that enables cross-institutions and interdisciplinary research.

The Semantic Web, coined by web inventor Tim Berners Lee, is normally deemed as the next generation of web. It aims at providing much more advanced integration capabilities that allow data, particularly the meaning of the data, to be shared and reused across application, enterprise, and community boundaries. The approach is to encode data semantics in a more formal and explicit way to make data machine-understandable and automatically consumable. A number of

---

This book started before Yimin Wang joined Lilly Singapore Centre for Drug Discovery; therefore, this book does not contain any content or opinion related to Lilly Singapore Centre for Drug Discovery and its parent company Eli Lilly & Co.

new web languages such as RDF (the Resource Description Framework) and OWL (the Web Ontology Language) have been proposed and standardized by the World Wide Web Consortium (W3C). These languages provide new capability for resource description and knowledge representation going far beyond the content presentation capabilities of HTML language and data tagging capabilities of the XML language. These languages can be used to define meaning of data, describe meta-data, represent terminologies and vocabularies, describe the input/output and functional capability of web service, and so forth.

As a synthesis of knowledge and web technologies and after over 10 years of development, the Semantic Web has been gaining tremendous attention worldwide. One noteworthy movement is the increasing recognition of this new technology in typical scientific areas such as life science,[1] earth science, and social science. For example, the World Wide Web Consortium (W3C) has established a Semantic Web interest group to focus on health care and life science (HCLSIG). This group is designed to improve collaboration, research and development, and innovation adoption of Semantic Web technologies in the health care and life science domains, aiding decision-making in clinical research and helping bridge many forms of biological and medical information across institutions. Other examples include the National Virtual Observatory, which is exploring the use of Semantic Web technologies for linking numerous astronomical resources together, and the FUSION project which is to apply, extend, and combine Semantic Web technologies and image analysis techniques to develop a knowledge management system to optimize the design of fuel cells.

As the semantic technology has been gaining momentum in various e-Science areas, it is important to offer semantic-based methodologies, tools, and middleware to facilitate scientific knowledge modeling, logical-based hypothesis checking, semantic data integration and application composition, integrated knowledge discovery, and data analyzing for different e-Science applications. However, such a research area does not yet exist in a coherent form. Influenced by the Artificial Intelligence community, the Semantic Web researchers have largely focused on formal aspects of logic languages or general-purpose semantic application development, with inadequate consideration of requirements from specific scientific areas. On the other hand, general science researchers are growing ever more dependent on the web, but they have no coherent agenda for exploring the emerging trends on Semantic Web technologies. It urgently requires the development of a multidisciplinary field to foster the growth and development of e-Science applications based on the semantic technologies and related knowledge-based approaches. We regard it as necessity to set a research agenda at this point in time, in order to steer the development and the research efforts in the most rewarding direction toward a common goal of realizing the semantic e-Science.

---

[1] Three recent relevant special issues published in life science area:
(1) *BMC Bioinformatics*: Semantic e-Science in biomedicine.
(2) *Journal of Biomedical Informatics*: Special issue on Semantic BioMed Mashup.
(3) *Journal of Web Semantics*: Special issue on Semantic Web in life science.

This book surveys the state of the art in the field of applying semantic technologies in typical e-Science applications. We anticipate it to provide a sufficient overview and valuable summary on semantic e-Science applications and to report the cutting-edge research outcome in this field. The book is conceived from a series of events.[2] We received nearly 30 submissions as a result of our public call for chapters, from which we selected 8 as the candidate chapters.

In Chapter 1, Prof. David De Roure and Carole Goble introduce the field being known as the 'Semantic Grid' that emphasizes on the Semantic Web and on the joined-up infrastructure to support the increasing scale of data, computation, collaboration, and automation as science and computing advance. Since its instigation in 2001 the Semantic Grid community has established a significant body of work on the role of the ideas and technologies of the Semantic Web in providing and utilizing the infrastructure for science.

In Chapter 2, Dr. M. Scott Marshall and colleagues introduce the Virtual Laboratory for e-Science (VL-e) project, which is a project with academic and industrial partners where e-Science has been applied to several domains of scientific research. The authors explain what 'semantic disclosure' means and how it is essential to knowledge sharing in e-Science. The authors describe several Semantic Web applications and how they were built using components of the proposed application toolkit.

In Chapter 3, Dr. Hock Beng Lim and colleagues describe a smart e-Science cyber-infrastructure for cross-disciplinary scientific collaborations. The proposed infrastructure forms the key resource sharing backbone that enables each participating scientific community to expose their sensor, computational, data, and intellectual resources in a service-oriented manner, accompanied by the domain-specific knowledge and semantic descriptions.

In Chapter 4, Dr. Alexander Garcia and colleagues propose a new methodology for developing ontologies within a decentralized setting in the e-Science context. The Melting Point (MP) approach is the product of direct first-hand experience and observation of biological communities of practice in which some of the authors have been involved.

In Chapter 5, Dr. Amitava Biswas and colleagues investigate semantic technologies for searching in e-Science grids. They present a survey of meaning representation and comparison technologies, followed by a design of meaning representation and comparison technique which is coherent to the cognitive science and linguistics models. This meaning comparison technique discerns complex meaning while enabling search query relaxation and search key interchangeability.

In Chapter 6, Dr. Enrico Pontelli and colleagues present a semantic e-Science system called *BioService Integration System (BSIS)*. BSIS is aimed at automating bioinformatics tasks through intelligent workflow construction. It provides a

[2] Four recent events:
(1) WWW2008 Semantic Web for health care and life sciences workshop.
(2) WWW2007 Workshop on health care and life science data integration for the Semantic Web.
(3) AAAI2007 Workshop on Semantic e-Science.
(4) ASWC2006 Workshop on Semantic e-Science.

graphic-based workflow language that can enable the description of workflows composed of abstract Semantic Web services and references to biologically relevant data. The workflows constructed in BSIS can be instantiated through automated planning techniques and automatically executed by adapting the *Web Service Integration Framework (WSIF)*.

In Chapter 7, Mikhail Simonov and Flavia Mazzitelli introduce the applications of rules and semantics in near-miss detection in nursing. Nursing science deals with human-to-human interaction delivering care service, showing several peculiarities compared with other life science domains. Semantic technologies can be applied to clinical nursing, where the risk management forms a particular application class and requires error detection and semantic technologies complementing best nursing strategies. The chapter investigates on possible risk control measures in above-said nursing, suggesting a combination of Information and Communication Technologies (ICT) and knowledge technologies.

In Chapter 8, Dr. Peisheng Zhao and colleagues describe a semantic e-Science application in the earth science domain. The Sensor Web provides an interoperable way to discover, task, and retrieve sensor data over the network. This chapter leverages recent advances in autonomous control and the Semantic Web technologies to present a systematic approach in which the Sensor Web and Earth science models are annotated with semantic meta-data and chained together in an autonomous way as a service chain to simulate the process of Sensor Web mining based on semantic inference. This approach significantly advances sensor data exploration with semantics in an interoperable way to improve the accuracy and timeliness of monitoring and predicting rapidly changing Earth phenomena.

In Chapter 9, Vít Nováček and colleagues investigate the application of semantic technologies in scientific publication. It describes the CORAAL system for knowledge-based life science publication search. It presents the approach from three different perspectives: (1) extraction of asserted publication meta-data together with the knowledge implicitly present in the respective text; (2) integration, refinement, and extension of the emergent content; (3) release of the processed content via a meaning-sensitive search and browse interface catering for services complementary to the current full-text search.

In summary, the merits of the Semantic Web with respect to scientific research are exhibited in different perspectives encompassing collective knowledge acquisition, semantic data integration, scientific workflow management and composition, integrative knowledge discovery and data mining, logic-based hypothesis checking, and so forth. The semantic e-Science is still an evolving area with rapid development. Although this book cannot give a complete account on all issues and topics, we hope it can shed some light on the most important aspects. We also hope it can push and promote the semantic technologies within various science fields, particularly in interdisciplinary scientific research.

Hangzhou, China — Huajun Chen
Immunos, Singapore — Yimin Wang
New Haven, Connecticut — Kei-Hoi Cheung

Contents

Contributors

Pieter W. Adriaans Informatics Institute, University of Amsterdam, Amsterdam, The Netherlands

Amitava Biswas Department of Computer Science, Texas A&M University, College Station, TX, USA, amitabi@cs.tamu.edu

Huajun Chen College of Computer Science, Zhejiang University, Hangzhou 310027, CN, China, huajunsir@zju.edu.cn

Kei-Hoi Cheung Yale Center for Medical Informatics, Yale University, New Haven, CT 06520-8009, USA, kei.cheung@yale.edu

Oscar Corcho Ontology Engineering Group, Departamento de Inteligencia Artificial, Facultad de Informtica, Universidad Politecnica de Madrid, Campus de Montegancedo, Madrid, Spain, ocorcho@fi.upm.es

David De Roure School of Electronics and Computer Science, University of Southampton, Southampton, UK, dder@ecs.soton.ac.uk

Liping Di Center for Spatial Information Science and Systems (CSISS), George Mason University, Fairfax, VA 22030, USA, ldi@gmu.edu

Alexander Garcia University of Bremen, Bremen, Germany, alexgarciac@gmail.com

Leyla Jael Garcia E-Business and Web Science Research Group, Bundeswehr University, Munich, Germany, leyla.garcia@ebusiness-unibw.org

Frank Gibson Abcam plc, Cambridge CB4 0WN, UK, frank.gibson@abcam.com

Carole Goble School of Computer Science, University of Manchester, Manchester, UK, carole.goble@manchester.ac.uk

Tudor Groza Digital Enterprise Research Institute (DERI), National University of Ireland, Galway, Ireland, tudor.groza@deri.org

Siegfried Handschuh Digital Enterprise Research Institute (DERI), National University of Ireland, Galway, Ireland, siegfried.handschuh@deri.org

Mudasser Iqbal Intelligent Systems Center, Nanyang Technological University, Nanyang Avenue, Singapore, mmiqbal@ntu.edu.sg

Sophia Katrenko Informatics Institute, University of Amsterdam, Amsterdam, The Netherlands

Hock Beng Lim Intelligent Systems Center, Nanyang Technological University, Nanyang Avenue, Singapore, limhb@ntu.edu.sg

Phillip Lord School of Computing Science, Newcastle University, Newcastle upon Tyne, UK, phillip.lord@ncl.ac.uk

Rabi Mahapatra Department of Computer Science, Texas A&M University, College Station, TX, USA, rabi@cs.tamu.edu

M. Scott Marshall Informatics Institute, University of Amsterdam, Amsterdam, The Netherlands, marshall@science.uva.nl

Flavia Mazzitelli Università di Torino, Facoltà di Medicina e chirurgia, Turin 10138, Italy, flavia.mazzitelli@tiscali.it

Edgar Meij Informatics Institute, University of Amsterdam, Amsterdam, The Netherlands

Suneil Mohan Department of Computer Science, Texas A&M University, College Station, TX, USA, suneil@cs.tamu.edu

Vít Nováček Digital Enterprise Research Institute (DERI), National University of Ireland, Galway, Ireland, vit.novacek@deri.org

Kieran O'Neill Terry Fox Laboratory, Vancouver, Canada, koneill@bccrc.ca

Yu Pan Department of Computer Science, New Mexico State University, Las Cruces, NM, USA, ypan@cs.nmsu.edu

Enrico Pontelli Department of Computer Science, New Mexico State University, Las Cruces, NM, USA, epontell@cs.nmsu.edu

Marco Roos Informatics Institute, University of Amsterdam, Amsterdam, The Netherlands, roos@science.uva.nl

Mikhail Simonov ISMB, Turin 10138, Italy; Politecnico di Milano, Dipartimento di energia, Milano 20133, Italy, simonov@ismb.it

Robert Stevens Department of Computer Science, University of Manchester, Manchester M13 9PL, UK, robert.stevens@manchester.ac.uk

Son Cao Tran Department of Computer Science, New Mexico State University, Las Cruces, NM, USA, tson@cs.nmsu.edu

Willem Robert van Hage Business Informatics, Faculty of Sciences, Vrije Universiteit, Amsterdam 1081HV, The Netherlands

Wenqiang Wang Intelligent Systems Center, Nanyang Technological University, Nanyang Avenue, Singapore, wqwang@ntu.edu.sg

Yimin Wang Lilly Singapore Centre for Drug Discovery, Immunos, Singapore 138648, Singapore, wangyimin@lilly.com

Yuxia Yao Intelligent Systems Center, Nanyang Technological University, Nanyang Avenue, Singapore, yxyao@ntu.edu.sg

Genong Yu Center for Spatial Information Science and Systems (CSISS), George Mason University, Fairfax, VA 22030, USA, gyu@gmu.edu

Peisheng Zhao Center for Spatial Information Science and Systems (CSISS), George Mason University, Fairfax, VA 22030, USA, pzhao@gmu.edu

About the Editors

Huajun Chen received his BS from the Department of Biochemical Engineering, and PhD from the College of Computer Science, both from Zhejiang University. At present, he serves as an associate professor in the College of Computer Science at Zheijiang University and was a visiting researcher at the School of Computer Science, Carnegie Mellon University. He is currently working for the China 973 'Semantic Grid' initiative and is the leader of the e-Science DartGrid semantic grid project.

Yimin Wang is an associate information consultant in Lilly Singapore Centre for drug discovery. He is currently leading projects related to Semantic Web R&D in the division of Integrative Computational Science to support drug discovery research. Before joining Lilly Singapore, he was a research associate at the Institute of Applied Informatics and Formal Description Methods (AIFB), University of Karlsruhe. He received his MS in 2005 after studying advanced computer science in the Medical Informatics Group at the University of Manchester, supervised by Prof. Alan Rector.

Dr. Kei-Hoi Cheung is an associate professor at the Yale Center for Medical Informatics. He received his PhD degree in computer science from the University of Connecticut. Since his PhD graduation, Dr. Cheung has been a faculty member at the Yale University School of Medicine. Dr. Cheung has a joint appointment with the Computer Science Department and Genetics Department at Yale. Dr. Cheung's primary research interest lies in the area of bioinformatics database and tool integration. Recently, he has embarked on the exploration of Semantic Web in the context of life sciences (including neuroscience) data and tool integration. Dr. Cheung edited *Semantic Web: Revolutionizing Knowledge Discovery in the Life Sciences* (Springer) and served as the chair of the *First International Workshop on Health Care and Life Sciences Data Integration for the Semantic Web*, which was held cooperatively with the WWW2007 conference. He was the guest editor of the special issue 'Semantic BioMed Mashup,' *Journal of Biomedical Informatics*. Dr. Cheung is also an invited expert to the Semantic Web Health Care and Life Science Interest Group launched by the World Wide Web Consortium.

Chapter 1
Supporting e-Science Using Semantic Web Technologies – The Semantic Grid

David De Roure and Carole Goble

Abstract The capabilities of the Web have had a very significant impact in facilitating new practice in science: it supports wide-scale information discovery and sharing, facilitating collaboration and enabling widespread participation in digital science, and increasingly it also provides a platform for developing and delivering software and services to support science. In this chapter we focus on the role of the ideas and technologies of the Semantic Web in providing and utilising the infrastructure for science. Our emphasis on Semantic Web and on the joined-up infrastructure to support the increasing scale of data, computation, collaboration and automation as science and computing advance has led to this field being known as the "Semantic Grid". Since its instigation in 2001 the Semantic Grid community has established a significant body of work, and the approach continues to underpin new scientific practice in multiple disciplines.

1.1 Introduction

As science progresses we witness evolution and revolution in scientific understanding. Coupled intricately with this is an evolution and revolution in scientific techniques and methods – the problem-solving ability of science advances as new methods lead to new understanding, and this in turn leads to new methods. In the last 10 years science has experienced a step change in problem-solving ability brought about by the increasing digitisation and automation of scientific instruments and practice. This has in part been achieved through the engagement between scientists and the providers of the advanced information, computational and software techniques that they need. We call this e-Science.

This revolution in technique and method has partly come about due to the deluge of data from new experimental methods and the rapidly evolving high-performance

D. De Roure (✉)
School of Electronics and Computer Science, University of Southampton, Southampton, UK
e-mail: dder@ecs.soton.ac.uk

H. Chen et al. (eds.), *Semantic e-Science*, Annals of Information Systems 11,
DOI 10.1007/978-1-4419-5908-9_1,

computing infrastructure used to process it [1]. In addition there are many technologies available to deliver the information infrastructure and software tools to support the increasingly digital and increasingly multidisciplinary practice of researchers. They involve tools for data, computation and collaboration, and their success depends on recognising and respecting the needs and practices of the researchers who use them [2].

The capabilities of the Web in particular have had a very significant impact in facilitating the practice of science. The Web supports wide-scale information discovery and sharing, facilitating collaboration and enabling widespread participation in digital science. Increasingly it also provides a platform for developing and delivering software and services to support science. More recently, the social practices of the Web are also influencing the social process of science, a movement characterised as Science 2.0 [3].

In this chapter we focus in particular on the role of the ideas and technologies of the Semantic Web in providing and utilising the infrastructure for science. Our emphasis on Semantic Web and on the joined-up infrastructure to support science has led to this field being known as the "Semantic Grid". Since its instigation in 2001 [4] the Semantic Grid community has established a significant body of work [5, 6] and the approach continues to underpin new scientific practice in multiple disciplines. Broadly categorised as "Semantic e-Science", Semantic Grid is distinguished by its emphasis on supporting the increasing scale of data, computation, collaboration and automation as science and computing advances.

We commence by reviewing the vision of e-Science and then discuss the technological challenges that underpin it and how they can be addressed through Semantic Web and "service-oriented science". We then look at Semantic Grid architecture and go on to illustrate how various Semantic Grid projects have demonstrated solutions to some of the challenges. We close by looking at Services, Semantics and Society – our evolving vision of e-Science.

1.2 The e-Science Vision

The UK e-Science programme was instigated in 2000 by John Taylor, then director general of UK Research Councils. e-Science was defined as being "about global collaboration in key areas of science and the next generation of infrastructure that will enable it" [7]. e-Science undertook to change the dynamic of the way science is undertaken, and this emphasis on science as well as infrastructure meant that success would ultimately be measured by new scientific outcomes.

We can trace the infrastructural insights further back. The distributed computational infrastructure was anticipated back in the 1960s by J. C. R. Licklider, an early architect of the Internet. Larry Roberts recalls: "Lick had this concept of the intergalactic network . . . everybody could use computers anywhere and get at data anywhere in the world. He didn't envision the number of computers we have today by any means, but he had the same concept – all of the stuff linked together throughout the world, so that you can use a remote computer, get data from a remote

computer, or use lots of computers in your job" [8]. While the Web has thus far grown largely around information access and provision, Licklider's vision also captures the computational aspect that underpins "in silico science" – using lots of computers to do your job.

Another key proposition of the Internet for science was the ability to achieve shared access to specialist facilities. This was captured in Wulf's 1989 vision of the collaboratory as "... a center without walls, in which researchers can perform their research without regard to geographical location, interacting with colleagues, accessing instrumentation, sharing data and computational resources, and accessing information in digital libraries" [9]. A collaboratory was seen as a means to more effective use of investments in facilities and also recognised that interaction between researchers is crucial to scientific progress.

The e-Science programme in the UK and cyberinfrastructure in the USA have been hugely significant because they provided the investment to realise this infrastructural potential. The Blue-Ribbon Advisory Panel on Cyberinfrastructure recognised the value in sharing the "commonalities", such as services, that underpin scientific applications: "Cyberinfrastructure makes applications dramatically easier to develop and deploy, thus expanding the feasible scope of applications possible within budget and organizational constraints, and shifting the scientist's and engineer's effort away from information technology development and concentrating it on scientific and engineering research. Cyberinfrastructure also increases efficiency, quality, and reliability by capturing commonalities among application needs, and facilitates the efficient sharing of equipment and services" [10].

We see then that e-Science envisages an infrastructure which provides a high degree of easy-to-use and seamless automation enabling flexible collaborations and computations on a global scale. It facilitates remote access to shared resources which may include facilities, data and services, together with support for the collaborative process of science. Furthermore, success involves not just building an infrastructure but providing the tools and techniques that enable it to be used to achieve scientific progress.

1.3 Semantic Grid: Service-Oriented Science

The Semantic Grid report, first published in 2001 [11], identified a set of technologies to deliver this e-Science vision. These included Web Services, Grid and Semantic Web.

Services rapidly emerged as a key abstraction in e-Science, providing an architectural model to achieve the vision of global collaboration and sharing of resource – in many ways, e-Science is service-oriented science. The World Wide Web Consortium (W3C) created the Web Services activity in 2002 to provide a standard means of interoperating between different software applications, running on a variety of platforms. As Web Services gained significant traction in industry and commerce – in Service-Oriented Architectures (SOA) – several projects in the UK e-Science programme successfully adopted this approach.

The Grid also evolved to provide a service-oriented architecture, focused on the special needs of e-Science. Ian Foster and Carl Kesselman described the Grid as being "distinguished from conventional distributed computing by its focus on large-scale resource sharing, innovative applications, and, in some cases, high-performance orientation" [12]. Ultimately the Grid community established its own Web Services architecture, known as the Open Grid Services Architecture (OGSA) [13]. Grid services allow a client to discover more information about the state and execution environment of a service, for example, this enables the monitoring required in some applications.

Web Services underpin the vision of the future service-oriented architecture for e-Science in which there are very large numbers of available services and an essential degree of automation in their use. Hence they are described in a machine-processable way, using the Web Services Description Language (WSDL), an XML grammar for describing network services as collections of communication endpoints capable of message exchange. WSDL definitions enable automation in both discovery and use of Web Services. Web Services themselves (and hence Grid Services) run over SOAP (originally defined as the "Simple Object Access Protocol"), a protocol for exchanging structured information in XML, typically using a Remote Procedure Call (RPC) model. Most commonly HTTP is used as an application layer transport for SOAP, providing message negotiation and transmission which is effective through proxies and firewalls.

The service orientation that we aspire to in the e-Science infrastructure involves dynamic composition of services. This is captured well in the Open Group Service Integration Maturity Model (OSIMM) which provides a set of milestones that measure progress towards SOA, defining seven levels of maturity: silo, integrated, componentised, simple services, composite services, virtualised services and dynamically reconfigurable services. Many legacy systems may have an emerging SOA (simple services); we seek composite, virtualised and dynamically reconfigurable services. We note that the Virtual Organisations (VOs) of Grid computing are relatively persistent and resourceful and have logically centralised, membership-oriented management structures based on existing trust relationships, as opposed to the dynamic business VOs that we seek [14].

It is also important to note that there is a widespread alternative service-oriented approach to distributed systems using HTTP, namely the REST (Representational State Transfer) architecture which underpins the Web and was defined by Fielding [15]. In REST, the methods which can act on a resource are predefined to be those supported by HTTP (i.e. GET, POST, PUT, DELETE), and instead of adding further methods, new resources are introduced. This fundamentally different architectural approach –sometimes called *Resource-Oriented Architecture* [16] – has enabled the Web to scale; e.g. the Web infrastructure is highly optimised for the GET operation through extensive caching. It also provides simplicity of programming and has wide support in programming and scripting languages.

To address the increasing scale and variety of services as we move into the future we need mechanisms for service discovery and we need automated mechanisms for establishing agreements between multiple parties. One approach to the latter

is provided by the WS-Agreement specification, an extensible XML language for specifying the nature of an agreement together with agreement templates to facilitate discovery of compatible agreement parties [17]. The Semantic Grid report also highlighted the techniques of multiagent systems to provide mechanisms for automatic selection of the combinations of services required to tackle a task [18].

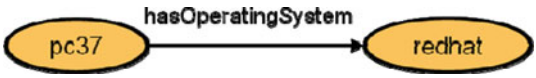
1.4 Semantic Web Essentials

In addition to services, the Semantic Grid report proposed Semantic Web as a key enabling technology, for both information and knowledge. The Semantic Web brings linked data in the same way as the Web has given us linked documents: "The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation" [19]. Hence Semantic Web standards provide us with mechanisms for integration and combination of data drawn from diverse sources. Semantic Web is also one approach to working with knowledge, as presented in [5].

e-Science is very much about data and about being "joined up" in the context of distributed collaborations, so there is a natural match with Semantic Web, a potential which is well recognised [20]. But here we see a relationship deeper in the infrastructure, because the vision of sharing of services and resources requires us to discover and use remote resources, and to do so in an automated way – imagine the future with a vast number of services and vast amount of data, which must be discovered and assembled to support the scientific task at hand. This vision of scale and automation inevitably demands machine-processable descriptions, and Semantic Web gives us exactly that. Hence, in addition to using Semantic Web to link up the data being processed by applications, the Semantic Grid vision also includes the description of resources and services – we push down into the infrastructure.

One technology that underpins both these propositions is the Resource Description Framework (RDF). RDF consists of statements called "triples", each of which contains a subject, property and object. For example, we can express that a particular computer (pc37) runs a particular operating system (redhat Linux) with the statement

```
< pc37, hasOperatingSystem, redhat >
```



Here we add two more statements, about pc38:

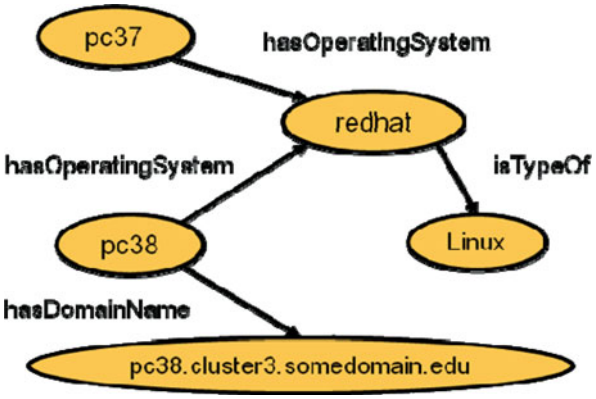
```
< pc38, hasOperatingSystem, redhat >
< pc38, hasDomainName, pc38.cluster3.somedomain.edu >
```

The subject, property and object are actually expressed as URIs, so the object "redhat" in the first two triples is represented by the same URI. Significantly, the

property “hasOperatingSystem” is also a URI, so the property is also something that can be agreed and shared.

The subject of one statement can be the object of another, so collections of RDF statements form a directed, labelled graph. Hence we can also add a statement about redhat:

```
< redhat, isTypeOf, linux >
```



Equivalently we could express these in a table:

Subject	Property	Object
pc37	hasOperatingSystem	redhat
pc38	hasOperatingSystem	redhat
pc38	hasDomainName	pc38.cluster3.somedomain.edu
Linux	isTypeOf	Linux

Immediately we can see how RDF provides machine-processable metadata. Standard metadata schema can be expressed in RDF by providing predefined properties, for example, the Dublin Core Metadata Initiative (DCMI) specifies a set of properties such as dc:creator, dc:title, dc:date. We can also see how RDF can enable accumulation of shared knowledge – we can make statements about any resources, and these are interlinked by shared identifiers so that others can interpret them.

There are some standard ways of expressing relationships between things on top of RDF. For example, we may wish to express that Windows and Unix are operating systems, XP and VISTA are subclasses of Windows and Linux is a subclass of Unix which in turn has redhat and Debian among its subclasses. RDF Schema (RDFS) provides us with this sort of structure, providing standard mechanisms for describing groups (classes) of related resources and the relationships between them. It also defines the concept of subproperty: if a property P is a subproperty of property P’, then all pairs of resources which are related by P are also related by P’.

Meanwhile, concept schemes such as thesauri, classification schemes and taxonomies are supported by the Simple Knowledge Organization System (SKOS), which classifies resources in terms of broader or narrower and allows preferred and alternate labels for concepts. The SKOS data model enables existing systems to be imported into the Semantic Web framework and makes it easy to develop new organisation systems.

Figure 1.1 depicts the layering of the Semantic Web. For more sophistication in expressing relationships and being able to use logic to make deductions, the Web Ontology Language (OWL) provides more vocabulary and a formal logic, Description Logic. One of the attractions of OWL is that it can be used in a simple way but can also extend to represent large and complex ontologies; it comes in three levels: OWL Lite, OWL DL (Description Language) and OWL Full. OWL can express much more than RDFS, such as the behaviour of properties (e.g. transitive, symmetric, reflexive, irreflexive, symmetric, functional, inverse functional). OWL also has a significant role in joining things up, for example, it provides owl:sameAs: to indicate that two URIs refer to the same concept.

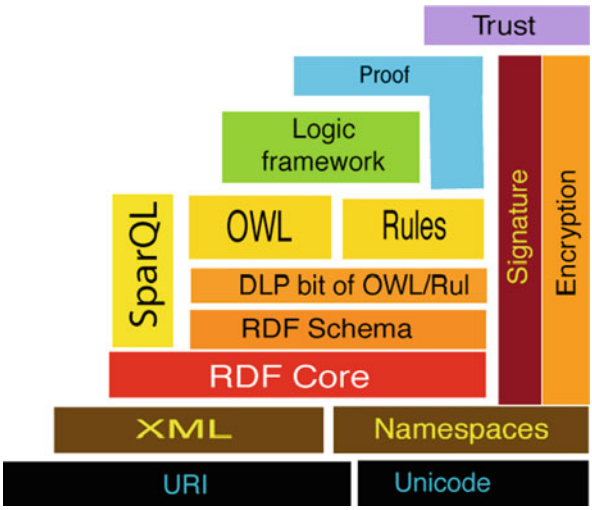


Fig. 1.1 The Semantic Web layer cake

The tooling to support these Semantic Web standards is now quite extensive. RDF is often published directly, and it can also be added to existing content by using microformats, which reuse HTML attributes and elements, or adding RDF statements directly into XHTML using RDFa. One of the most popular tools is the RDF triplestore, which stores and manages RDF triples and enables them to be queried. Many triplestores are available and in terms of scale are now measured in billions of triples [21]. The standard means of querying a triplestore is SPARQL (SPARQL Query Language for RDF) which contains capabilities for querying required triple patterns, conjunctions, disjunctions and optional patterns. A "SPARQL endpoint"

provides a machine-friendly interface to query an RDF store, and we envisage that they will be a key architectural abstraction for the future Semantic Grid.

Through these technologies, entire datasets are now being exposed in RDF. The Linking Open Data initiative recommends best practice for exposing, sharing and connecting datasets in this way, exposing them through SPARQL endpoints. For example, Bio2RDF is an open source project which has created a network of coherent linked data across life sciences databases by normalising external URIs during the conversion from the original format to RDF – a "Semantic web atlas of postgenomic knowledge". Each converted database has a SPARQL endpoint, and all documents can be retrieved using REST. Figure 1.2 illustrates the public data sources available in the "Linked Data Cloud".

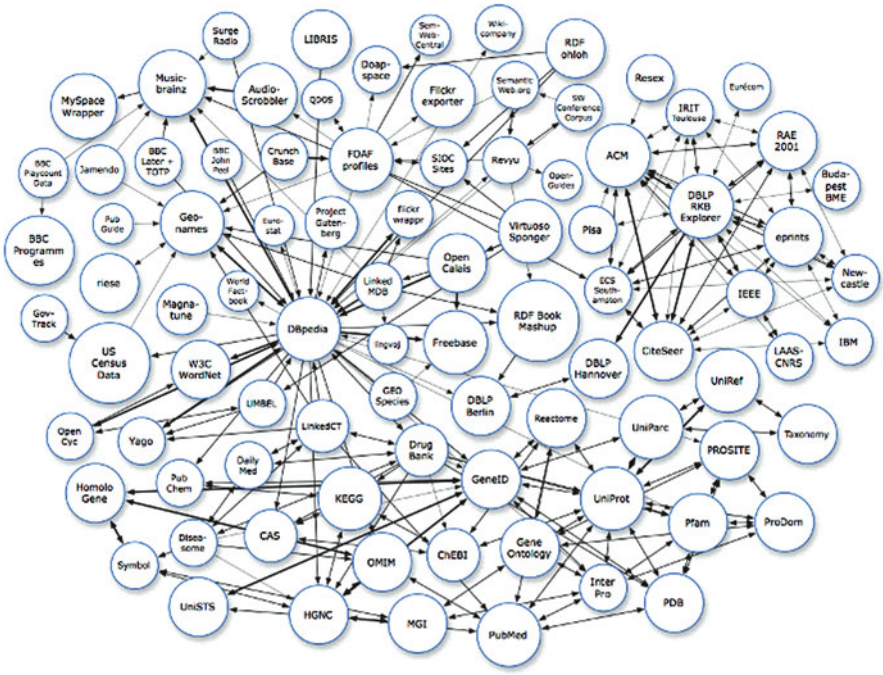


Fig. 1.2 The Linked Data Cloud, as of March 2009

1.5 Achieving the e-Science Vision

How exactly do these technologies help? First we need to understand the requirements. The 2005 follow-up Semantic Grid paper identified the requirements of the e-Science infrastructure under 12 headings [6]:

1. Resource description, discovery, and use;
2. Process description and enactment;

3. Autonomic behaviour;
4. Security and trust;
5. Annotation;
6. Information integration;
7. Synchronous information streams and fusion;
8. Context-aware decision support;
9. Communities;
10. Smart environments;
11. Ease of configuration and deployment;
12. Integration with legacy IT systems.

In the following sections we consider several of these in turn and discuss how they are addressed through the Semantic Grid approach.

1.5.1 Resource Description, Discovery, and Use

The system must store and process potentially huge volumes of distributed content in a timely and efficient fashion, perhaps through the federation of resources, and hence be able to identify content, services, computational resources, Grid-enabled instruments, and so on. It must be able to discover and locate these resources efficiently and to negotiate access. It also requires generation and processing of job descriptions and on-demand and dynamically planned use of resources in order to meet quality of service requirements and achieve efficient resource utilisation.

This requirement is addressed by the machine-processable metadata. We have four important observations based on our experience in Semantic Grid projects:

1. Metadata has a life cycle, like data. It is created, consumed and needs to be maintained, preserved and curated. It is important to attend to the metadata fabric of e-Science and recognise that it is as much part of the infrastructure as data itself.
2. The type and quality of metadata needed to support discovery is different to that which is needed for automated use. Discovery metadata can help us identify a set of candidate services or resources, but acquisition and utilisation then require different information about the specific service or resource.
3. It is not essential to store everything in RDF: it can suffice to be able to import and export in RDF format. However, it is useful to adhere to RDF principles and practice with respect to shared identifiers in order to achieve those benefits – to be indistinguishable from everything being stored in RDF. Sometimes the best way to achieve this is indeed to store data in RDF, perhaps by shadowing an alternative store.
4. Often there is confusion between the data model and serialisations of that model. A model expressed in RDF might be serialised in many different ways. Conversely, just because something is serialised in RDF does not mean it has an RDF data model.

1.5.2 Process Description and Enactment

To support the creation of virtual organisations of services, the system needs descriptions (such as workflows) to facilitate composition of multiple resources and mechanisms for creating and enacting these in a distributed manner.

This is an important requirement: it takes us to "level 7" of the maturity model and complements the e-Science focus on data with reusable descriptions of how that data is processed. These descriptions enhance the derived data by providing a record of its creation and provenance. They also enable processes to be replayed, repeated and reused [22].

Two key approaches have emerged for describing assemblies of services:

1. *Scientific workflows*. A scientific workflow is the description of a process that specifies the co-ordinated execution of multiple tasks so that, for example, data analysis and simulations can be repeated and accurately reported. Alongside experimental plans, Standard Operating Procedures and laboratory protocols, these automated workflows are one of the most recent forms of scientific digital methods and one that has gained popularity and adoption in a short time [23]. They represent the methods component of modern research and are valuable and important scholarly assets in their own right.
2. *Mashups*. A mashup is a Web application that combines data from one or more sources, typically achieved by a script which accesses open APIs. Mashups are also characterised by rapid development, typically achieved using REST principles. The most popular examples of mashups are perhaps those that make use of Google Maps to add location information to other data. Like workflows, mashups can be shared and reused.

Figure 1.3 illustrates the Taverna workflow system [24]. The workflow is described by a graph which is created in a workflow editor and then enacted with a workflow enactment engine. Workflows and mashups can both be used to provide new services. Workflows can arguably provide a more abstract and declarative description of process which facilitates reuse.

Workflow systems benefit from Semantic Web technologies for discovery, planning and use of services; for recording the provenance of derived data; and ultimately for describing and sharing workflows themselves. We discuss this further in the next section. It is anticipated that mashups will also benefit from Semantic Web technologies, in discovering resources, services and previous mashups and in integrating data that is semantically enriched [25].

1.5.3 Autonomic Behaviour

Systems should auto-configure to meet the needs of their multiple users in dynamically changing circumstances and "self-heal" in the presence of faults (thus, the systems will appear to be reliable but, in practice, they may conceal failures and

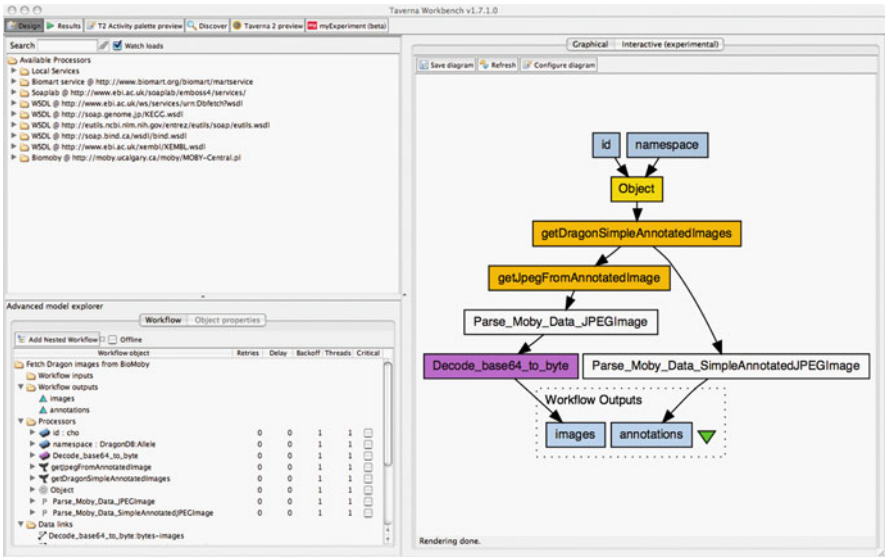


Fig. 1.3 The Taverna workflow workbench

exception handling at various levels). The system should also support evolutionary growth as new content and services become available.

We see this behaviour first in delivering services and second in using them. For example, cloud storage services provide a robust service through simple APIs, using SOAP or REST, and hide the fault-tolerance mechanisms that are used in delivering that service: some of the autonomic behaviour occurs in the cloud infrastructure, which is designed to cope with the inevitable failure of components. Meanwhile above the service level we need to deal with failure of services: lack of availability, failure during execution, incorrect results or even malicious behaviour. Late-binding of workflows to service instances, a failover mechanism with alternate services and compensation mechanisms are established techniques to cope with faults above the service level. In the future we anticipate a role for multiagent systems negotiation techniques in achieving the required autonomic behaviour.

1.5.4 Security and Trust

There are authentication, encryption and privacy requirements, with multiple organisations involved, and a requirement for these to be handled with minimal manual intervention. Indeed from one perspective this is what really defines virtual organisations. Related to this are issues of agreements and accounting: different stakeholders need to be able to retain ownership of their own content and processing capabilities, allowing others access under the appropriate terms and conditions. By their nature, policies need to be represented such that they can be applied to multiple resources and with consistent interpretation.

Semantic Web provides the mechanism for linking together data sources which have different owners. It also gives us a means to represent policy and rules, for example, through the creation of policy ontologies. Representing usage logs in RDF promises to be a powerful mechanism to facilitate their interpretation, manually or through use of automated tools.

1.5.5 Annotation

From logging a sample through to publishing the analysis, annotations enrich the description of any digital content and enhance reuse. Annotation also supports the important notion of provenance, whereby sufficient information is stored to repeat the experiment, reuse the results or provide evidence that this information was indeed produced at this time (the latter may involve a third party). Ideally, in many cases, annotations will be acquired automatically. Obtaining annotations is only half the story however; we also need to make use of them. Examples of such usage include finding papers, finding people, finding a previous experimental design (these queries may involve inference), annotating the uploaded analysis and configuring a smart space to suit its occupants. Annotation may thus be distributed and collaborative.

Annotation adds value to existing information, and we have seen previously that RDF is designed for exactly this purpose: we can create RDF statements about any resources. The source of annotation may be explicit actions of users – such as tagging resources using a Web page – or annotations can be collected automatically through usage. For example, the choice of a service can be informed by the record of its previous performance, which is effectively an automatic annotation of the service. By recording that a data source has been used by a certain workflow, other users of that data source can find the workflows (and hence services) that have used it or the secondary data derived from it.

1.5.6 Information Integration

The ability to make meaningful queries over disparate information stores, and to make use of content in ways which may or may not have been anticipated, requires interoperability of information. For example, this may involve mapping between terminologies used in different domains. This is the classical role of Semantic Web technologies.

In discussing resource descriptions we have focused on metadata, but the Linked Data activity demonstrates that RDF is effective for data representation too. This is a field where ontologies have particular potential because they provide means by which heterogeneous resources can be integrated. Research in this area predates e-Science, dating back to knowledge representation, and has addressed important challenges in complexity and scalability, in reasoning algorithms and in the tools for developing and using ontologies. It is an important aspect of Semantic e-Science and explored comprehensively in later chapters.

1.5.7 Communities

Communities of practice need to be formed, maintained and disbanded, perhaps with their own membership criteria and rules of operation. This involves identifying the set of individuals in a virtual organisation through collaborative tools and exploiting knowledge about communities of practice across disciplines. This is important for everything from expert-finding to assembling the large teams needed in multidisciplinary research endeavours such as climate change.

The representation of information about people, and of the social network, is well established in the Semantic Web using the Friend-of-a-Friend (FOAF) vocabulary which describes individuals, their activities and their relationships to other people and objects. More recently, the Semantically Interlinked Online Communities (SIOC) ontology is gaining acceptance for integration of online community information. Meanwhile, Google's *OpenSocial* defines a common API for social applications across multiple Web sites.

1.6 A Semantic Grid Approach to In Silico Bioinformatics

The first of our two Semantic Grid case studies is the myGrid project (www.mygrid.org.uk).

The myGrid Consortium is a multi-institutional, multidisciplinary research team focusing on the challenges of e-Science. Formed in 2001, myGrid has produced the Taverna Workflow Management System (www.taverna.org.uk) which includes a GUI-based rich-client workbench, an enactment engine for server and client deployment based on an extended lambda calculus [26] and knowledge management services for provenance [27, 28] and discovery [29]. Over 3,500 publicly available services are currently accessible as workflow components. Taverna is very widely used and is the cornerstone of many of the consortium's projects and the focus of many collaborations.

myGrid is designed for openness and ease of extension: it is a loosely coupled, extensible set of components that can be adopted independently by tool developers but which are designed to work together in a distributed service-oriented architecture. They work together by being organised conceptually around two communication buses into which myGrid's software services are plugged, as shown in Fig. 1.4. The use of two buses effectively decouples the business of creating the experimental environment from the business of managing the rich e-Science content.

The Experiment Interoperation "bus" is an event-enabled communication infrastructure that couples together during the running of the software:

- Common core myGrid services for creating, deleting, publishing and managing data and metadata. These are part of the software suite and not the actual databases or analytical tools that will form steps of the workflows;

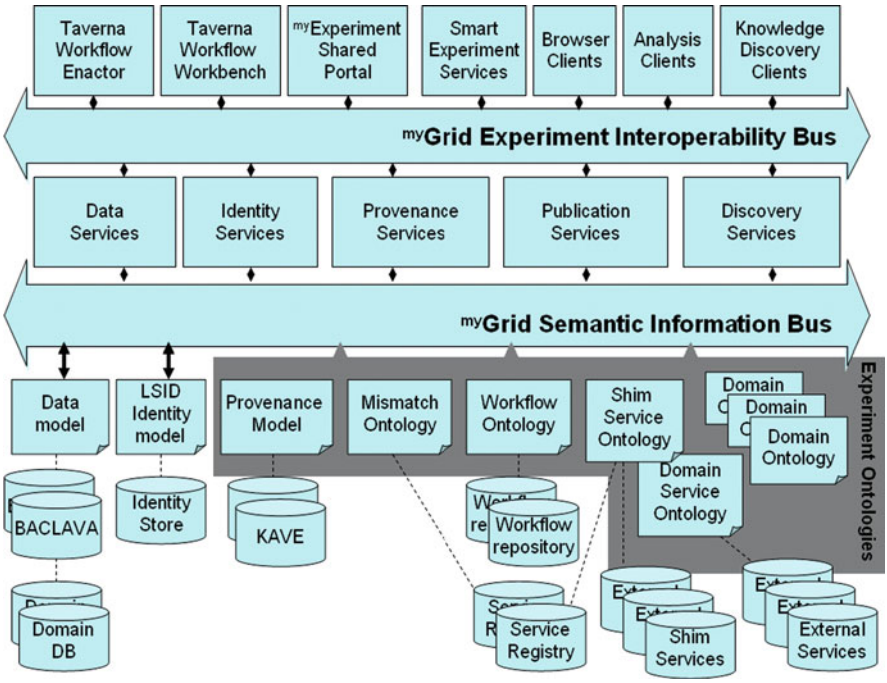


Fig. 1.4 The myGrid middleware: Joining up in silico experiments with semantics

- The core Taverna workflow enactor for running experiments that interact with the core myGrid services;
- Core myGrid clients such as the Taverna workbench, which has its own plug-in architecture (i.e. new functions can easily be added using programs which comply with its software interfaces), and provenance browsers;
- Domain-specific and external client-side applications that use those services and clients, for example, the Utopia sequence visualisation application [30].

The Semantic Information "bus" carries the persistent semantic and data content and models that the core services share, provide and consume, for example, the provenance model and service discovery model. The information flowing on this bus is the annotated experimental data. On the one side are the services and even the client-side applications that can tap into this semantic infrastructure; on the other side are the services for creating and storing the semantic content. The Semantic bus is the Semantic infrastructure for myGrid and can be seen as a Semantic Web of experiments.

Taverna refers to both a workbench and a workflow enactment environment that can be run separately from the workbench by a third-party application such as

Utopia. The Taverna workbench is a myGrid application that services and clients plug into and allows the scientist to design and run workflows. The myExperiment Web-based collaborative environment [31] allows the scientist to organise, communicate, publish and share their experiments and their outcomes. Taverna provides and consumes the Semantic Web of experiments for an individual scientist; myExperiment provides an environment for exploring and contributing to a wider Semantic Web of science.

The myGrid team provides the ontologies and keeps a registry of the services and makes available a repository of workflows that the scientist can draw upon or use their own. The domain services that the workflow enactor invokes are separate from the myGrid software suite and are hosted by their own service providers and registered in the Biocatalogue [22]. The provenance and data results are stored locally with the scientist or they can configure shared stores as they wish; they are not held centrally in a resource owned by myGrid.

This architecture is designed to support the flows of semantic information within the scientific process. Our experience of building and using myGrid effectively provides an evaluation of the promise of the Semantic Web against the real requirements of the Life Scientist.

1.7 A Semantic Datagrid for Chemistry

In our second case study we focus on the linking of data. The CombeChem project (combechem.org) set out to achieve a complete end-to-end connection between the chemistry laboratory bench and the intellectual chemical knowledge that is published as a result of an experiment – this is described as "publication at source" [32]. Underlying this is the crucial observation that the details of the origins of data are just as important to understanding as their actual values. Publication at source describes the need to capture data and its context from the outset – it is much easier to collect this information at source and at creation than attempt to add it later.

The creation of original data in the laboratory is accompanied by information about the experimental plan that was being followed, a record of what actually occurred and the laboratory conditions. There then follows a chain of processing such as aggregation of experimental data, selection of a particular data subset, statistical analysis or modelling and simulation; some of the steps in the process require significant computing resources. The handling of this information may include annotation of a diagram or editing of a digital image. All of this generates secondary data, accompanied by the information that describes the process that produced it, and this might be maintained in a variety of distinct datastores. Through the principle of publication at source, all this data is made available for subsequent reuse in support of the scientific process, subject to appropriate access control. Figure 1.5 illustrates the plan and experimental record captured as an RDF graph.

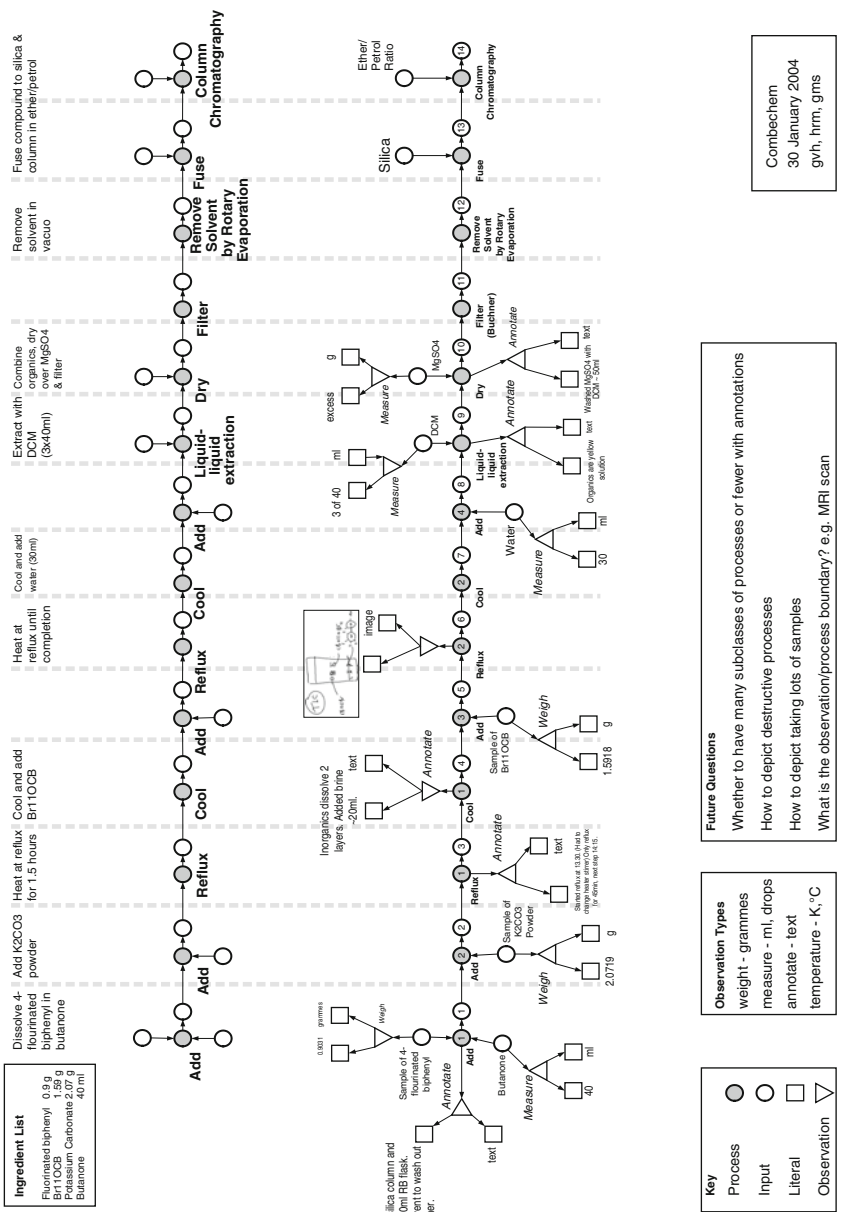


Fig. 1.5 Example of an experimental plan and process record

Our design approach adopted five principles [33]:

1. Grounding in established operational practice – our starting point was to study chemists at work;
2. Capturing a rich set of associations between all types of things, expressed pervasively in RDF and hence explicitly addressing the sharing of identifiers;
3. Metadata capture should be automated as far as possible;
4. Information would be reused in both anticipated and unanticipated ways;
5. The storage, maintenance and transport of metadata will be given equal consideration to data, ensuring availability of accurate metadata, a dependable provenance record and comprehensive understanding of the context of data.

The system supports the chemist through the whole life cycle of an experiment, broken down into four parts, with the "PPPP" mnemonic: plan, perform, ponder and publish. Although simplistic, this does capture many of the aspects of the discovery process.

The acquisition starts with planning and performing, using the smart laboratory and Grid-enabled instrumentation [34]. By studying chemists within the laboratory, Electronic Laboratory Notebook technology has been introduced to facilitate the information capture at this earliest stage [35]. Additionally pervasive computing devices are used to capture live metadata as it is created at the laboratory bench, relieving the chemist of the burden of metadata creation. This aspect, which is a significant enabler for publication at source, is set to grow considerably as pervasive computing deployment advances.

It is significant that this capture makes use of both a record of the researcher's planned activity and what actually occurs. In the UK the chemist has to produce a plan of the experiment as a list of the reagents to be used, and any associated hazards, as part of the COSHH (Control Of Substances Hazardous to Health) assessment. The plan is a key part of the knowledge capture in support of the publication at source model. It also enables the Electronic Laboratory Notebook to provide a guide to the experiments in the laboratory. Capturing the experiment as a reusable digital artefact also facilitates sharing and reuse of experiment design.

Experimental results are then used by researchers within CombeChem's grid-based information and knowledge sharing environment that integrates existing chemical knowledge, specifically structure and property data sources. The research that is conducted, which may involve simulations using the Grid, leads to new results and to publication. At the outset we anticipated that we would support this environment by using RDF as a means of integrating across the many established data sources, which include relational databases and third-party information providers. This is a good example of the use of RDF triplestores in conjunction with database solutions.

In practice we found that the chemistry researchers were keen to import chemical information directly into the RDF stores. The benefits were the uniform description and the flexible schema afforded by this approach, contrasting the diversity of relational databases where changing schema was impossible or achievable only at

very high cost. This triplestore contains tens of millions of RDF triples and represents a substantial Semantic Web deployment. The chemical data was obtained from a range of publicly available databases including the ZINC database [36], the National Institutes for Health (NIH) and in particular the National Cancer Institute (NCI) chemical data. We used the open source 3store triplestore software, which was used in a similar harvesting role in the CS AKTive Space project [37].

The current target is hundreds of millions of triples. However, CombeChem has moved away from managing everything in one scalable triplestore. This harvesting and hoarding approach to the "mashup" benefits the immediate users but itself is not in the spirit of the open approach to publishing knowledge. Rather, the knowledge sources are made available in RDF and imported into triplestores as required – the Web itself then becomes the scalable triplestore, and the sources are available for reuse. The ontologies that were created to support this environment are described in [38].

As an example of the output of this process, the eCrystals interface shown in Fig. 1.6 provides a Web page complete with a 3D visualisation of a molecule, data collection parameters and links back to the files of data which led to this output (see ecrystals.chem.soton.ac.uk). Behind this simple interface there is a complex picture including a diverse set of stakeholders – the federation model involves data collection, data curation and preservation in databases and databanks, institutional data

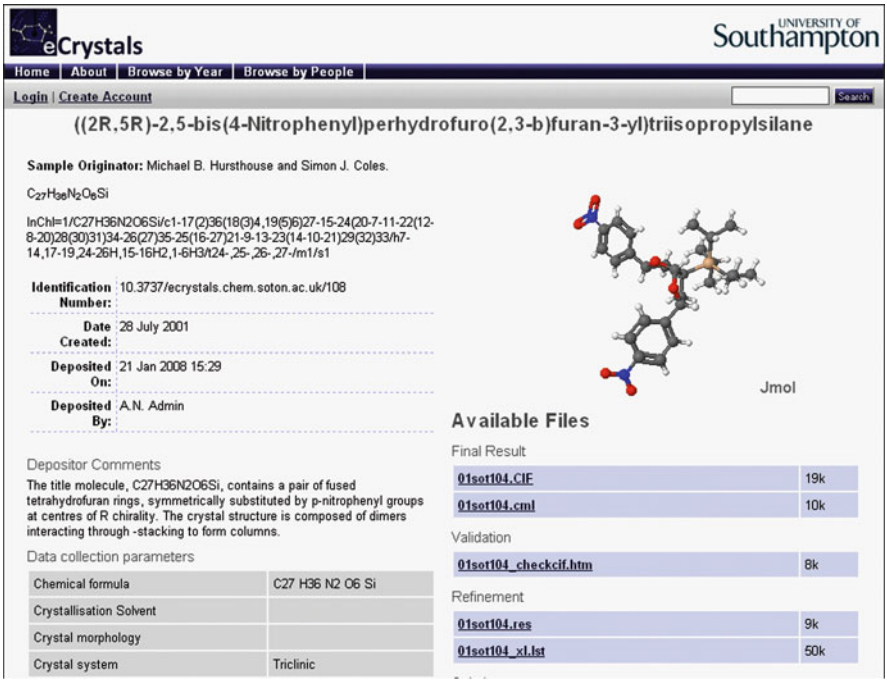


Fig. 1.6 The eCrystals interface

repositories, aggregator services, portals and publishers. The CombeChem team also produced an academic paper in this form [39].

The interlinking of research data and research publication is the subject of the eBank project, which provides open access crystallography data interlinked with its derived research publications – it is possible to chase back to see exactly where results have come from or even to find research publications arising from data. In line with the digital library context for this work, OAI (Open Archives Initiative) metadata is harvested from institutional data repositories. As part of this exercise, the Repository for the Laboratory (R4L) project is developing digital data and document repositories for laboratory-based science (see r4l.eprints.org). R4L addresses the interactions between repositories of primary research data, the laboratory environment in which they operate and repositories of research publications they feed into.

The CombeChem Semantic Datagrid demonstrates how an RDF triplestore can be used to provide enhanced recording, storage and retrieval of scientific data, in a flexible fashion. The triplestores contain the rich metadata that describes the relationships within the scientific information, and the data that is described may be held in a variety of existing stores. Since the metadata is machine-processable, it provides the necessary basis for sophisticated querying and for automation in information processing, which could, for example, include curation. The analysis of the complex data and provenance information needed for chemical information provides valuable lessons for representation and handling of the necessary level of detail involved with data in other sciences.

The CombeChem project created an ontology for units. For example, when we state that a solution has a strength of 0.02 mol dm^{-3} then 0.02 is the value of the measurement, mol is the first unit (indicating one Avogadro's constant of molecules), dm^{-3} is the second unit ("per decimetre cubed") which can be further decomposed into deci (an SI prefix for 1/10th), metre (the SI unit for length) and -3 (the exponent of the preceding unit, present if not equal to one).

1.8 Architectures for the Semantic Grid

In Europe the 6th Framework Programme (2002–2006) funded many projects in Advanced Grid Technologies, Systems and Services, depicted in Fig. 1.7, and convened the *Next Generation Grids Expert Group* (NGG). Many of the projects involve Semantic Grid in some form. In this section we will focus on one of these projects, OntoGrid, and on a report produced by the experts group which looked into the future of this approach.

1.8.1 S-OGSA – A Reference Architecture for the Semantic Grid

The OntoGrid project set out to produce the technological infrastructure for the rapid prototyping and development of knowledge-intensive distributed open services for the Semantic Grid, observing that designing applications that make use of

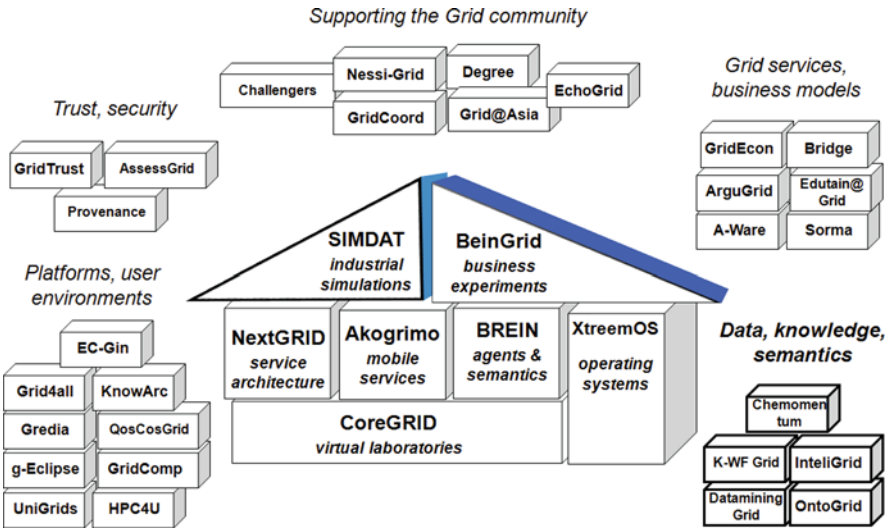


Fig. 1.7 The "Grid House" of projects funded in the European 6th framework programme

a Semantic Grid requires a new and sound methodology. The project used two case studies to develop grid systems that optimised cross-process, cross-company and cross-industry collaboration. Significantly, OntoGrid produced Semantic-OGSA (S-OGSA), the first reference architecture for the Semantic Grid. S-OGSA is depicted in Fig. 1.8 and is guided by six general design principles which are reported in [40]:

1. *Parsimony of architectural elements*. The architectural framework should be as lightweight as necessary and should minimise the impact on legacy Grid infrastructure and tooling. It should not impose the vocabulary or the structure to be used in the semantic descriptions, since these will be application or middleware dependent, though a basic set of reusable vocabularies can be provided, related to different aspects of the model.
2. *Extensibility of the framework*. Rather than defining a complete and generic architecture, define an extensible and customisable one.
3. *Uniformity of the mechanisms*. Semantic Grids are Grids, so any S-OGSA entity included in the architecture will be OGSA-observant. Similar to the Grid resources they are associated with, knowledge and metadata should exhibit manageability aspects. Semantic descriptions could have state and soft state characteristics – they have a lifetime and may change during their life. S-OGSA must encapsulate both stateless and stateful Grid services, as OGSA does. Knowledge services in S-OGSA are OGSA-observant Grid services; for instance, metadata stores and ontology services are just special kinds of data services.

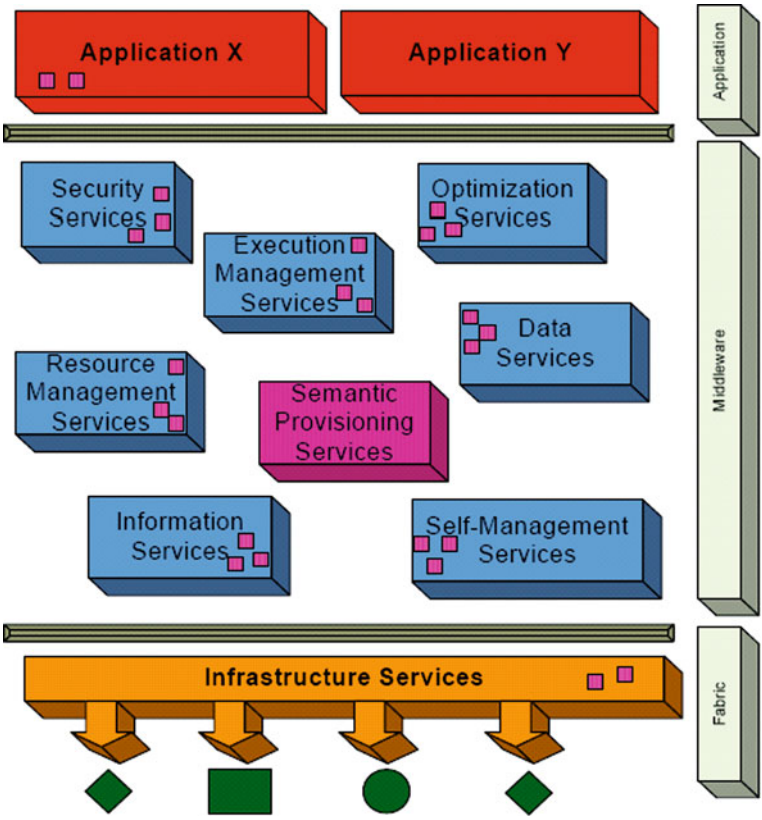


Fig. 1.8 A simplified view of the architecture proposed by OGSA, with the additional components from S-OGSA (in the centre and in adjacent boxes)

4. *Diversity of semantic capabilities*. A dynamic ecosystem of Grid services ranging over a spectrum of semantic capabilities should coexist at any one time. Grid entities do not need to be Semantic Grid entities. Semantic capability may be possible for some Grid resources all of the time, and maybe all Grid resources some of the time, not all resources all of the time. Entities in the Semantic Grid are thus classified as ignorant of the explicit semantics associated with another entity, aware that another entity has explicit associated semantics but incapable of processing it or aware that another entity has explicit associated semantics and capable of processing it, partially or completely.
5. *Heterogeneity of semantic representation*. Any resource's property may have many different semantic descriptions, and each of them may be captured (or not) in different representational forms (text, logic, ontology, rule).
6. *Enlightenment of services*. Services should have a straightforward migration path that enables them to become knowledgeable. The cost involved in the migration to the Semantic Grid must be minimised in order to improve the impact and

uptake of Semantic Grid and to take advantage of current tooling and services. Thus S-OGSA should have minimal impact on adding explicit semantics to current Grid entity interfaces or on Grid services that are ignorant of Semantic Grid entities; Grid entities should not break if they can consume and process Grid resources but cannot consume and process their associated semantics, and if a Grid entity understands only part of the knowledge it consumes it should be able to use it as a best effort (that is, there are different degrees of awareness and semantic processing capabilities). During their lifetime, Grid entities can incrementally acquire, lose and reacquire explicit semantics.

1.8.2 The Service-Oriented Knowledge Utility

The *Next Generation Grids Expert Group* (NGG) also identified the need for semantically rich facilities and a service-oriented approach. In the last quarter of 2005 the group convened to identify the gaps between the leading-edge of Grid technologies and the end-user. The convergence of the evolving NGG vision with the service-oriented vision of significant European industry stakeholders in NESSI (Networked European Software and Services Initiative, an industry-led Technology Platform that aims to provide a unified view for European research in Services Architectures and Software Infrastructures) naturally led the NGG group to define the scientific and technological requirements necessary to evolve Grids towards the wider and more ambitious vision of *Service-Oriented Knowledge Utilities* (SOKU). Figure 1.9 shows the influences on SOKU.

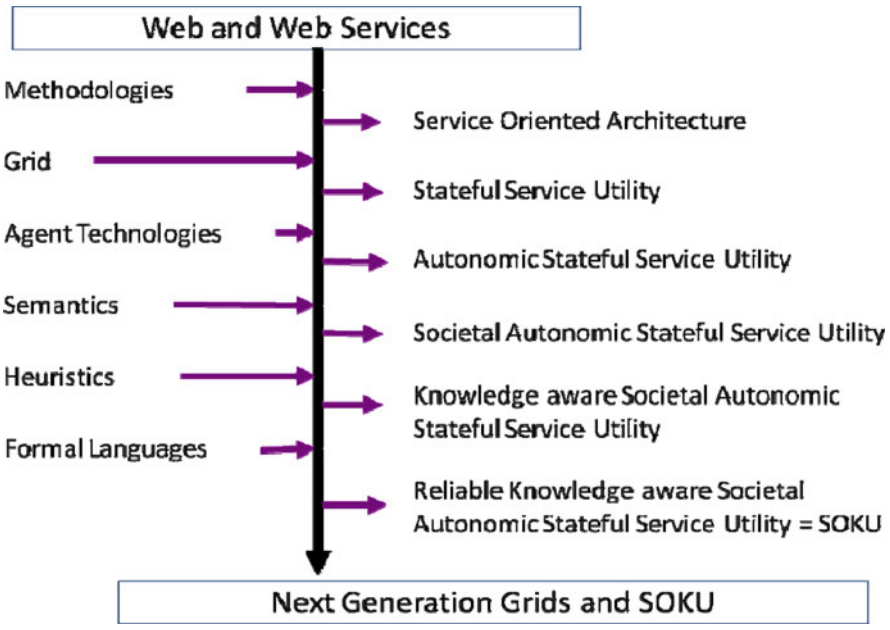


Fig. 1.9 Influences on SOKU

The confluence of the Service-Oriented Knowledge Utility paradigm and Next Generation Grids is a compelling approach to the future IT architecture. It encompasses several important domains including foundations of service-oriented computing, service-oriented architecture, grid and utility computing, business process management and, business integration. In order to realise the vision a number of challenging research topics were identified:

1. Life cycle management: on-the-fly service creation and deployment; robust, efficient and semantically aware discovery of services; composition of services; management of functional and non-functional properties and requirements; and support for multiple "economy models" for the grid.
2. Trust and security: ad hoc and managed virtual organisations of digital and physical entities; policy and business practice; service-level agreements; authentication and authorisation in a multi-domain environment in which entities have multiple identities and multiple roles.
3. Adaptability, dependability and scalability: self-systems; peer-to-peer; scalability.
4. Raising the level of abstraction: higher level programming models and tools; new or improved management abstractions; better operating systems capable of managing more complex resources and requirements from application, service and system contexts; abstract/virtual service containers; compact data formats.
5. Pervasiveness and context awareness of services: high-level interoperability, smooth composition and automatic self-organisation of software with structure and behaviour changing at run time; non-functional requirements related to interoperability, heterogeneity, mobility and adaptability.
6. Underpinning semantic technologies: mechanisation of composition; scalable reasoning and formalisation; heterogeneous and dynamic semantic descriptions; life cycle of knowledge; collaboration and sharing.
7. Human factors and societal issues: user requirements and evaluation; intersection between the physical world and the digital; personalisation techniques; issues of collaboration and community; socio-economic aspects.

1.8.3 The Semantic Grid Community

The Semantic Grid community has developed many interesting projects in a spectrum of application areas. The following examples give a flavour of the breadth of activity:

- PolicyGrid (http://www.policygrid.org/) is a UK e-Science project funded as part of the Economic and Social Research Council eSocial Science initiative. The project aims to support policy-related research activities within social science by developing appropriate Grid middleware tools which meet the requirements of social science practitioners. The vision of the Semantic Grid is central to the PolicyGrid research agenda.

- Semantic sensor grids for rapid application development for environmental management (http://www.semsorgrid4env.eu/) use Semantic Web and Web Services to develop an integrated information space where new sensor networks can be easily discovered and integrated with existing ones, together with the rapid development of flexible and user-centric environmental decision support systems that use data from multiple, autonomous, independently deployed sensor networks and other applications.
- The Networked Environment for Music Analysis (NEMA) project (http://nema.lis.uiuc.edu/) brings together world leaders in the domains of music information retrieval, computational musicology and e-humanities research to create an open and extensible resource framework based on Web Services and Semantic Web that facilitates the integration of music data and analytic/evaluative tools.
- The Data Grid Team at the Grid Technology Research Center in AIST, Japan, has developed a service-based middleware that utilises Grid and Semantic Web technology. The OGSA-DAI-RDF middleware extends OGSA-DAI (which allows data resources to be accessed via Web services) to provide access to RDF triplestores, with a query language interface based on SPARQL [41].

1.9 Moving Forward

In this chapter we have seen that Semantic Grid is about the semantic and service-oriented infrastructure of e-Science and e-Research. This emphasises data, metadata and services, but we have also seen that collaboration is fundamental to e-Science and hence there is also an important social dimension to the fabric of the Semantic Grid.

Our latest work builds on the experience of myGrid and CombeChem to major on that social infrastructure. The myExperiment project (http://www.myexperiment.org) is our social Web site for sharing scientific workflows and other artefacts of digital science. It supports a social network and makes it easy to create groups of people and to discover and share things. This approach makes it easier for people to find workflows, and it also enables community curation of workflows which in turn helps address the difficult problem of "workflow decay" due to the flux of the service landscape.

The architecture of myExperiment is shown in Fig. 1.10. The myExperiment data model is implemented in Ruby on Rails and is closely tied to the models for the user interface as shown. Furthermore due to the perpetual beta approach to agile management of the project, that data model is subject to change. We have therefore built a parallel server which is updated with the live content but has a stable data model expressed as a modularised ontology and provides a SPARQL endpoint. The ontology builds on Dublin Core, FOAF, SIOC and Object Reuse and Exchange (ORE).

The BioCatalogue (http://www.biocatalogue.org/) is a sister project which provides a registry of Web Service in the life sciences. The service descriptions are curated by the service providers, expert, users and automatic tools, again taking a

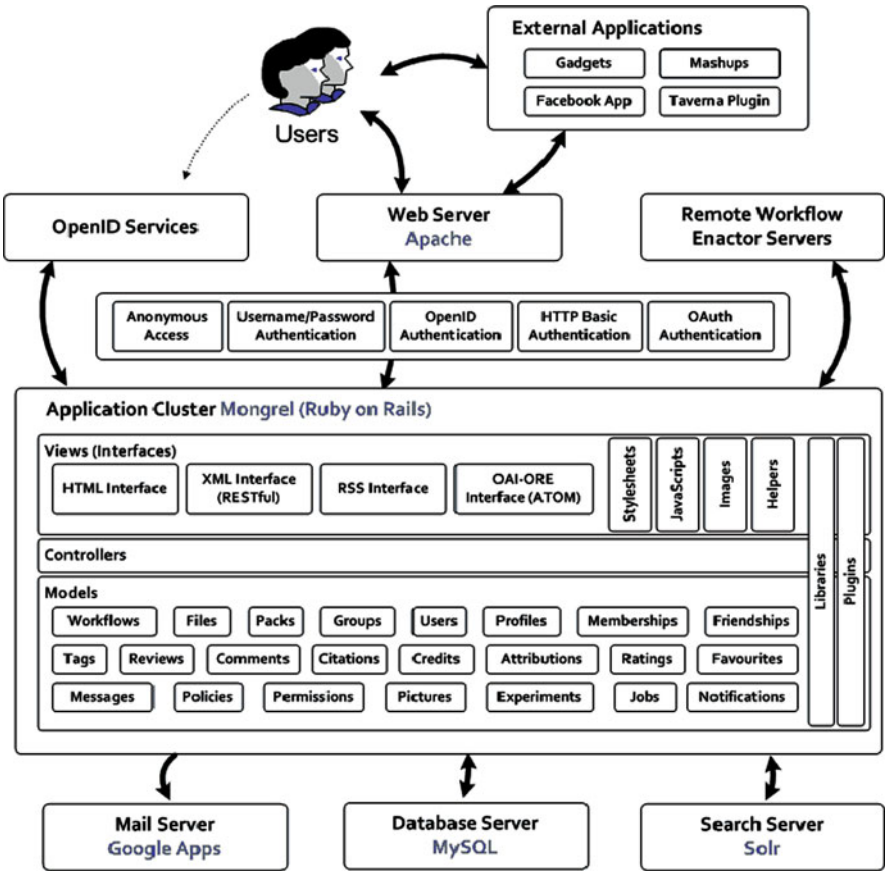


Fig. 1.10 The myExperiment architecture

social approach. Coupled with myExperiment, BioCatalogue can learn about service utilisation while myExperiment learns about services and their availability.

1.10 Summary

As we reflect over all the Semantic Grid activities, we observe three crucial aspects which will lead us forward:

1. Automation is necessary to cope with scale, and automation requires machine-processable metadata. At the time of writing, the SeekDa service registry contains around 30,000 services: we are not yet at the scale of services that make this critical but the trend is clear. At some degree of scale, solutions such as

workflows and mashups have benefit, but the challenge ahead is to deal with such a scale that we will need to enhance these solutions.
2. e-Science is about joining up resources and joining up people, to achieve new research outcomes. Again, the Semantic Web offers a solution, especially through the Linked Data initiative and activities like SIOC. All usage leads to annotation, and annotation adds value so that our resources can be used in ways that were anticipated but also, importantly, unanticipated. Hence Semantic Web can help tap the "wisdom of the crowd". In fact we anticipate that research will not always be shared via academic papers but rather by sharing reusable digital "Research Objects" which describe the bundles of resources that define our scientific endeavours.
3. In both cases the information fabric of e-Science comprises content, which has a life cycle of its own, and the infrastructure to support it. The Semantic Web tooling has reached a maturity which makes this viable, as demonstrated through the projects presented above, and best practice is becoming established.

Acknowledgments This chapter includes material based on information provided by the myGrid, CombeChem and OntoGrid teams. The authors wish to thank these teams and the Semantic Grid community for their assistance in this work.

References

1. Hey, T., Trefethen, A.: The data deluge: An e-science perspective. (2003) 809–824
2. De Roure, D., Goble, C.: Software design for empowering scientists. IEEE Software 26 (2009) 88–95
3. Shneiderman, B.: Computer science: Science 2.0. Science 319 (2008) 1349–1350
4. De Roure, D., Jennings, N.R., Shadbolt, N.R.: The semantic grid: A future e-science infrastructure. (2003) 437–470
5. Goble, C.A., DeRoure, D., Shadbolt, N.R., Fernandes, A.A.A.: Enhancing services and applications with knowledge and semantics. In: Foster, I., Kesselman, C. (eds.) The Grid 2: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, San Francisco (2004) 431–458
6. De Roure, D., Jennings, N.R., Shadbolt, N.R.: The semantic grid: Past, present, and future. Proceedings of the IEEE 93 (2005) 669–681
7. Taylor, J.: Talk given at UK e-Science Town Meeting in London (2001)
8. Segaller, S.: Nerds 2.0.1: A Brief History of the Internet, TV Books (1998)
9. National Research Council Committee on a National Collaboratory. National collaboratories: Applying information technology for scientific research. National Research Council, Washington, DC (1993)
10. Atkins, D.E., Droegemeier, K.K., Feldman, S.I., Garcia-Molina, H., Klein, M.L., Messerschmitt, D.G., Messina, P., Ostriker, J.P., Wright, M.H.: Revolutionizing Science and Engineering Through Cyberinfrastructure: Report of the National Science Foundation Blue-Ribbon Advisory Panel on Cyberinfrastructure, National Science Foundation (2003)
11. De Roure, D., Jennings, N.R., Shadbolt, N.R.: Research Agenda for the Semantic Grid: A Future e-Science Infrastructure in UK e-Science Technical Report Series, National e-Science Centre, Edinburgh, UK (2001)
12. Foster, I.: The anatomy of the grid: Enabling scalable virtual organizations. International Journal of Supercomputer Applications 15 (2001) 2001

13. Foster, I., Kesselman, C., Nick, J.M., Tuecke, S.: The physiology of the grid. (2003) 217–249
14. Surridge, M., Taylor, S., De Roure, D., Zaluska, E.: 2005. Experiences with GRIA — Industrial Applications on a Web Services Grid. In Proceedings of the First international Conference on E-Science and Grid Computing (December 05–08, 2005). E-SCIENCE. IEEE Computer Society, Washington, DC, 98–105 (2005). DOI= http://dx.doi.org/10.1109/E-SCIENCE.2005.38
15. Fielding, R.T.: Architectural Styles and the Design of Network-Based Software Architectures, University of California, Irvine, Irvine, CA (2000)
16. Richardson, L., Ruby, S.: RESTful Web Services, O'Reilly Media (2007)
17. Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Nakata, T., Pruyne, J., Rofrano, J., Tuecke, S., Xu, M.: Web Services Agreement Specification (WS-Agreement) in, Open Grid Forum, Grid Resource Allocation Agreement Protocol (GRAAP) WG (2007)
18. Foster, I., Jennings, N.R., Kesselman, C.: Brain Meets Brawn: Why Grid and Agents Need Each Other in Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems – Volume 1, IEEE Computer Society, New York, NY (2004)
19. Berners-Lee, T., Hendler, J., Lassila, O.: "The semantic web." Scientific American. (May 2001) 34–43
20. Hendler, J.: Communication: Enhanced science and the semantic web. Science 299 (2003) 520–521
21. Semantic Web Challenge http://challenge.semanticweb.org/, visited
22. Belhajjame, K., Goble, C., 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 Workshop Indianapolis, Indiana (2008)
23. Gil, Y., Deelman, E., Ellisman, M., Fahringer, T., Fox, G., Gannon, D., Goble, C., Livny, M., Moreau, L., Myers, J.: Examining the challenges of scientific workflows. Computer 40 (2007) 24–32
24. Oinn, T., Greenwood, M., Addis, M., Alpdemir, N., Ferris, J., Glover, K., Goble, C., Goderis, A., Hull, D., Marvin, D., Li, P., Lord, P., Pocock, M., Senger, M., Stevens, R., Wipat, A., Wroe, C.: Taverna: Lessons in creating a workflow environment for the life sciences. Concurrency and Computation: Practice and Experience 18 (2006) 1067–1100
25. Sheth, A.P., Gomadam, K., Lathem, J.: SA-REST: Semantically interoperable and easier-to-use services and Mashups. Internet Computing, IEEE 11 (2007) 91–94
26. Turi, D., Missier, P., Goble, C., De Roure, D., Oinn, T.: "Taverna workflows: syntax and semantics," e-science and grid computing, International Conference on, pp. 441–448. In: Third IEEE International Conference on e-Science and Grid Computing (e-Science 2007) (2007). http://doi.ieeecomputersociety.org/10.1109/E-SCIENCE.2007.71
27. Zhao, J., Wroe, C., Goble, C., Stevens, R. Quan, D., Greenwood, M.: Using semantic web technologies for representing e-science provenance. In: Proceedings of the 3rd International Semantic Web Conference, volume 3298, pages 92–106, Hiroshima, Japan (2004)
28. Missier, P., Belhajjame, K., Zhao, J., Roos, M., Goble, C.A.: Data lineage model for taverna workflows with lightweight annotation requirements. IPAW (2008) 17–30
29. Lord, P., Alper, P., Wroe, C., Feta, C.G.: A Light-Weight Architecture for User Oriented Semantic Service Discovery In Proceedings of Second European Semantic Web Conference, ESWC 2005, Heraklion, Crete, Greece, May–June 2005. pp. 17–31, Springer-Verlag LNCS 3532 2005
30. Pettifer, S.R., Sinnott, J.R., Attwood, T.K.: 2004. UTOPIA — user-friendly tools for operating informatics applications: Conference reviews. Comparative and Functional Genomics 5(1) (Feb. 2004) 56–60. DOI= http://dx.doi.org/10.1002/cfg.v5:1
31. De Roure, D., Goble, C., Stevens, R.: The design and realisation of the virtual research environment for social sharing of workflows. Future Generation Computer Systems 25 (2009) 561–567
32. Frey, J.G., Roure, D.D., Carr, L.: Publication At Source: Scientific Communication from a Publication Web to a Data Grid in EuroWeb 2002 Oxford (2002)

33. Taylor, K., Gledhill, R., Essex, J.W., Frey, J.G., Harris, S.W., De Roure, D.: 2005. A semantic datagrid for combinatorial chemistry. In: Proceedings of the sixth IEEE/ACM international Workshop on Grid Computing (November 13–14, 2005). International Conference on Grid Computing. IEEE Computer Society, Washington, DC, 148–155 (2005). DOI= http://dx.doi.org/10.1109/GRID.2005.1542736
34. Hughes, G., Mills, H., De Roure, D., Frey, J.G., Moreau, L., schraefel, M.C., Smith, G., Zaluska, E.: (2004) The semantic smart laboratory: a system for supporting the chemical e-Scientist, Organic Biomolecular Chemistry 2(2004) 3284. DOI: 10.1039/B410075A
35. schraefel, M.C., Hughes, G., Mills, H., Smith, G., Payne, T., Frey, J.: Breaking the book: Translating the chemistry lab book into a pervasive computing lab environment. In: CHI 2004, April 24–29, 2004, Vienna, Austria (2004)
36. Irwin, J.J., Shoichet, B.K.: ZINC – A free database of commercially available compounds for virtual screening. Journal of Chemical Information and Modeling 45(1) (2005) 177–182
37. Shadbolt, N., Gibbins, N., Glaser, H., Harris, S., schraefel, M.C.: "CS AKTive space, or how we learned to stop worrying and love the semantic web." IEEE Intelligent Systems 19(3) (May/June 2004) 41–47. Doi:10.1109/MIS.2004.8
38. Taylor, K.R., Gledhill, R.J., Essex, J.W., Frey, J.G., Harris, S.W., De Roure, D.C.: Bringing chemical data onto the semantic web. Journal of Chemical Information and Modeling 46(3) (2006) 939–952. http://dx.doi.org/doi:10.1021/ci050378m
39. Rousay, E.R., Fu, H.C., Robinson, J.M., Essex, J.W., Frey, J.G.: Grid-based dynamic electronic publication: A case study using combined experiment and simulation studies of crown ethers at the air/water interface. Philosophical Transactions Of the Royal Society of the London A 363 (2005) 2075–2095
40. Corcho, O., Alper, P., Kotsiopoulos, I., Missier, P., Bechhofer, S., Goble, C.: An overview of S-OGSA: A reference semantic grid architecture. Journal of Web Semantics 4 (2006) 81–154
41. Kojima, I.: Design and Implementation of OGSA-DAI-RDF in GGF16 3rd Semantic Grid Workshop, GGF (2006)

Chapter 2
Semantic Disclosure in an e-Science Environment

M. Scott Marshall, Marco Roos, Edgar Meij, Sophia Katrenko, Willem Robert van Hage, and Pieter W. Adriaans

Abstract The Virtual Laboratory for e-Science (VL-e) project serves as a backdrop for the ideas described in this chapter. VL-e is a project with academic and industrial partners where e-science has been applied to several domains of scientific research. Adaptive Information Disclosure (AID), a subprogram within VL-e, is a multi-disciplinary group that concentrates expertise in information extraction, machine learning, and Semantic Web – a powerful combination of technologies that can be used to extract and store knowledge in a Semantic Web framework. In this chapter, the authors explain what "semantic disclosure" means and how it is essential to knowledge sharing in e-Science. The authors describe several Semantic Web applications and how they were built using components of the AIDA Toolkit (AID Application Toolkit). The lessons learned and the future of e-Science are also discussed.

2.1 Introduction

2.1.1 Semantic Disclosure

Knowledge discovery lies at the heart of scientific endeavor. The discovery, storage, and maintenance of knowledge form the foundation of scientific progress. Consequently, if we define e-Science as "enhanced Science," then it is essential that e-Science should enhance knowledge discovery and re-use. Ultimately, e-Science is about discovering and sharing knowledge in the form of experimental data, theory-rich vocabularies, and re-usable services that are meaningful to the working scientist. Furthermore, the e-Scientist should be able to work with user interfaces that tap into knowledge repositories to provide familiar terminologies and

M.S. Marshall (✉)
Informatics Institute, University of Amsterdam, Kruislaan 403, Amsterdam, The Netherlands
e-mail: marshall@science.uva.nl

H. Chen et al. (eds.), *Semantic e-Science*, Annals of Information Systems 11,
DOI 10.1007/978-1-4419-5908-9_2,

associations from the domain of inquiry, unencumbered by data schemas, format conversion, or quirky user interfaces. For many scientists, there is the sense that if we could somehow work with conceptual icons and the terms that we are already using to think about a problem, we could more easily manipulate our ideas with the logic and rules of our own definition.

As steadily more organizations build data and knowledge repositories, there is a growing interest in information extraction and knowledge capture technologies. In order to make knowledge discovery possible, we must be able to access and harness existing knowledge. There are vast amounts of knowledge that are digitally available in both publications and on the Web. However, most knowledge is not available in a machine-readable form. The process of knowledge extraction, i.e., text mining from literature can provide us with knowledge distilled from scientific discourse. The same principle can be applied to text documents associated with a given type of data, where the knowledge extracted from the associated texts is used as a *semantic annotation* of the associated data resource. Whether performed manually or accomplished with the method that we've just described, the result is *semantic disclosure*, where meaning about a thing (i.e., resource) is disclosed in a machine-readable statement about it. The mined knowledge and associated data can then be reasoned about by new computational experiments to create new hypotheses and knowledge.

2.1.2 The Semantic Web

The *semantic stack* of the World Wide Web Consortium (W3C) was created in order to provide machine readable and interoperable knowledge exchange. The "semantic stack" is built on a set of standards that handle progressively more specific requirements. At the base is eXtended Markup Language (XML), which has provided a basis for data exchange by providing the representation, schema, and syntax of XML. On top of XML [1], the Resource Description Framework (RDF) provides a way to express statements in "triples" of "subject predicate object."[1] When the subject of one RDF statement unifies with the object of another, the statements connect together to form a graph or "web." The SPARQL Query Language for RDF, a 2008 W3C recommendation, enables query of the RDF graph to look for graph patterns. The modeling language RDF-Schema (RDF-S) provides a basis for hierarchies and subsumption reasoning ("dog isA mammal" and "mammal isA animal" implies "dog isA animal"). The Web Ontology Language (OWL) extends the basic class definitions of RDF to enable reasoning and modeling using description logic. A rule layer tops the semantic stack with the Rule Interchange Format (RIF). Although the semantic stack cannot handle all forms of knowledge, it provides a practical basis for

[1] In the case of *semantic disclosure*, the subject could identify a data or service resource in order to disclose something about it, such as its *dc:creator* ("*dc*" from the Dublin Core standard, see http://dublincore.org/documents/dces/).

interoperable storage, retrieval, and exchange of knowledge. This basis is supported by a variety of implementations, many of them freely available and open source. Additional W3C standards related to RDF are also available, including RDFa for embedding RDF in web pages and Friend of a Friend (FOAF) for social networks.

The Simple Knowledge Organization System (SKOS) for vocabularies makes it possible to relate concepts as "broader" or "narrower" in a way that is intuitive and useful for structured vocabularies. One of the most useful aspects of SKOS is that it enables forward chaining across relations skos:broader and skos:narrower for the induction of hierarchies without the requirements of the more strict logic of OWL. Modeling such relations with OWL properties can result in "ontological overcommitment," for example, where an instance that is assigned a class unintentionally inherits inaccurate properties. The more vague semantics of SKOS can be convenient for modeling "associations" or relations that are not only as well defined as in OWL or RDF-S (or not defined at all yet) but also appropriate for modelling-type hierarchies. We will discuss a few applications of SKOS in the remainder of this chapter.

A common misconception about the Semantic Web is that all knowledge must be first represented in a large ontology and that some sort of large knowledge network on the scale of the Web must exist before anyone can reap the benefits. However, Semantic Web technologies and tools are already being used today to effectively manage and exchange knowledge without requiring practitioners to develop an entire software infrastructure beforehand. As steadily more people follow Linked Open Data principles[2] and learn how to apply the semantic stack, more data is becoming available as interlinked RDF and a Semantic Web is gradually emerging. An excellent introduction to Semantic Web can also be found in the book *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL* [2].

2.1.3 Making Sense of the Digital Deluge

Many see biomedical science, with its wide spectrum of disciplines and corresponding variety of data, as the ideal proving ground for Semantic Web. Indeed, biologists seem keen to apply the new technologies being developed in the context of e-Science [3]. Understanding the human body, with its many layers of interwoven dynamic molecular systems that interact on subcellular, cellular, tissue, and organ levels may well be one of the most formidable challenges left to science. While research in the life sciences has provided us with additional knowledge about many biological phenomena, many of the mechanisms that are most essential to understanding disease remain mysteries. As an example, the gene for Huntington's Disease was the first genetic disease mapped to a chromosome with DNA polymorphisms in 1983 and isolated in 1993, yet the actual causal chain behind the progression from gene mutation to neurodegeneration is still unknown. Starting in 1991, the Human

[2] http://linkeddata.org/

Genome Project (HGP) sequenced the entire human genome, providing an initial reference sequence of human DNA in 2001. In 1993, around the time the HGP was getting started, the Web began its rapid expansion with the release of the Mosaic Web Browser. Public databases such as the GDB Human Genome Database soon became available on the Web, setting the stage for a new era of data sharing, public data curation, code sharing, bioinformatics, and computational biology. The journal *Nucleic Acids Research* now tracks more than 1,000 publicly accessible molecular biology databases [4].

2.1.4 Data Integration

A logical approach to such an abundance and variety of data is to combine data from several adjacent areas of research and look for patterns in the resulting aggregation. However, researchers hoping that the assembly of the new genome-scale data being amassed would bring new insight have experienced firsthand that the data integration of heterogeneous data is a non-trivial exercise and that scaling up only adds to the problem. Large data archiving projects have experienced similar problems, with researchers struggling to create the right data design and interface in order to avoid creating yet another "massive data graveyard."[3] Translational medicine, an effort to couple the results of fundamental life science research with clinical applications, has become a visible goal of large organizations such as the National Institute of Health (NIH) in the United States. The integration of data from "bench to bedside" (i.e., data from wet laboratory research domain to clinical domain called translational medicine) is a key socio-economic issue for the health care and pharmaceutical industries because it makes maximal use of data across multiple disciplines and enables direct knowledge sharing between disciplines. In order to reach across the boundaries of several disciplines, from life science to drug discovery and virtual screening of compounds, and from drug design to clinical treatment, translational medicine will require the bridging of many terminologies and a strong framework for data integration as its foundation.

2.1.5 W3C Semantic Web for Health Care and Life Sciences Interest Group

Semantic Web offers the means to perform data integration. Indeed, the W3C Semantic Web for Health Care and Life Sciences Interest Group[4] (HCLS IG) proposes that the necessary foundation for translational medicine goals could be provided by a set of practices that make use of W3C Semantic Web standards [5]. The HCLS IG got its start in 2004, when the W3C Workshop on Semantic

[3] David Shotton, University of Oxford.

[4] http://www.w3.org/2001/sw/hcls/

Web for Life Sciences[5] brought a large community of interested parties with 115 participants, resulting in a charter for the HCLS IG in early 2005. Many workshop participants were already performing pioneering research related to Semantic Web for HCLS and banded together in task forces to create technology demonstrations and document them. The HCLS IG was rechartered in 2008 for an additional 3 years to continue its mission to develop, advocate for, and support the use of Semantic Web technologies for biological science, translational medicine, and health care. The group has approximately 100 official participants at the time of writing, with a wide range of participation from industry and academia.

Within the "BioRDF" task force of the HCLS IG, a demonstration of data integration using Semantic Web was built and first shown at a WWW conference in Banff in 2007. The demonstration successfully answered a scientific question about Alzheimer's disease to show the value of being able to query across the data of 15 public databases from the Web. The data was first aggregated into an RDF repository with care to choose an OWL design that was in line with OBO Foundry Methodology. Many researchers contributed significantly to this effort as can be seen from the list of Contributors and Acknowledgements in the HCLS Interest Group Note [6]. Additional work also demonstrated the extension of the knowledge base with SenseLab data [7]. The resulting knowledge base has reached production level in the Neurocommons [8] and continues to be developed by the BioRDF task force, with instances running at DERI Galway and at Free University Berlin.

2.1.6 Semantic Architecture

An e-Science environment brings with it an expanded set of resources and possibilities. The familiar hardware-based set of resources known to system and middleware programmers such as CPU, memory, network bandwidth, input and output devices, and disk space are augmented by "soft" resources such as data, knowledge, and services. In both grid and web environments, service-oriented architecture (SOA) supplies the advantage of data and software components that are readily available on the network for spontaneous incorporation into applications. In fact, the sheer abundance of shared heterogeneous resources can become an impediment to use and the complex tooling necessary to deploy them tends to hinder the uninitiated developer. For an end user, the resources of interest could be things as diverse as journal articles, image data, mass spectrometry data, R scripts, services, workflows, and spreadsheets. How can the user discover and select the resources that are most appropriate to the task at hand? Many factors conflate to make a complex problem of matching requirements, preferences, and policies with the resources at hand. The resource discovery problem is present at many system levels. At the application level, users must discover the knowledge resources (e.g., vocabularies), applications, and parameters that are relevant to their particular task, in their particular application

[5] http://www.w3.org/2004/07/swls-ws.html

domain. At the application development and middleware level, developers must discover services and data, preferably in a way that can be automated, in order to dynamically adjust for variable resource availability and access. In fact, the problem of data and service discovery is common to many computing environments, not only grid but also the Web and large data repositories of many sorts. An e-Science scenario could involve resources from all of the above environments but the challenge remains the same: to manage heterogeneous resources from a single user interface.

2.1.7 The Virtual Laboratory for e-Science Project

The Virtual Laboratory for e-Science[6] (VL-e) is a project with academic and industrial partners where e-science has been applied to several domains of scientific research. Adaptive Information Disclosure[7] (AID), a subprogram within VL-e, is a multi-disciplinary group that concentrates expertise in information extraction, machine learning, and Semantic Web – a powerful combination of technologies that can be used to extract and store knowledge in a Semantic Web framework that enables effective retrieval via both keyword and semantic search. In order to support metadata and knowledge management, AID has created a set of web services as generic components that support the building of applications that are customized to a particular domain. The web services and the applications built around them comprise the AIDA Toolkit (AID Application Toolkit). The AIDA Toolkit and its applications have been developed in cooperation with project partners from several application domains and address a variety of use cases. The AIDA Toolkit has been applied to use cases in bioinformatics, medical imaging, and food informatics during the first 4 years of the VL-e project.

Several AIDA applications have been created that can be executed from four different interfaces: Taverna (workflows), a Taverna plugin, a web interface, and a Java application for accessing grid resources called the VBrowser. The AIDA Toolkit and its applications have been developed in cooperation with project partners from several application domains and address a variety of use cases. In the remaining sections, we will describe our experiences and the lessons learned while designing, building, and applying the AIDA Toolkit to use cases in bioinformatics, medical imaging, and food informatics during the first 4 years of the VL-e project. We will describe how we created a workflow for hypothesis support in biology through information extraction and how this leads to issues in computational experimentation such as the choice of knowledge representation that enables knowledge re-use and knowledge provenance, as well as the need to support semantic types in workflows. We will also discuss the quest for food terminologies that would be useful to our Food Informatics partners and how this finally led to a Web browser interface

[6] http://www.vl-e.nl

[7] http://adaptivedisclosure.org/

with access to customized (Lucene) indexes and multiple terminologies, among them a SKOS translation of a food ontology that was specially developed by the Food Informatics Consortium. The same combination of customizable indexes with structured vocabularies for search has also been applied in a Java application that provides access to grid resources, paving the way to perform semantic retrieval and annotation of grid resources. This Java application can be used by medical imaging researchers to manage image data that is stored and transported on the grid. We also describe how we extended it to semantically annotate and retrieve medical images.

2.2 The AIDA Toolkit

For the purpose of knowledge management, we would like to perform knowledge capture. In the context of a laboratory, knowledge capture can be regarded as the process of collecting and managing related knowledge resources, especially those related to an experiment. In this case, knowledge capture can be directly compared to resource management, where resource aggregation requires a way to associate disparate resource types and the creation of intuitive methods for retrieval. In a laboratory, the resource types can include various types of raw data, such as image data, as well as ontologies and vocabularies for annotation of those resources. In contrast, in a knowledge base, the emphasis is more on the collection of facts and rules than that of data, although links to the evidence on which the knowledge is based are generally desired. Of course, this practical distinction between knowledge capture in a laboratory and a knowledge base should eventually give way to a complete chain of evidence from data and data provenance to distilled fact.

In order to build and maintain a knowledge base, a knowledge engineer needs methods to extract, represent, and manipulate knowledge resources such as facts and rules. Ideally, round-trip knowledge engineering would be possible, where facts in the knowledge base could be directly extracted from the data, and knowledge base maintenance would consist of updating the evidence with new data in order to generate any new facts that would follow from it. The set of web services that we describe here are components of AIDA in a service-oriented architecture that cover the basic functionality necessary to create a knowledge management application. In particular, a user can combine them in a flexible way with other web services providing search, extraction, and annotation functionality.

The AIDA Toolkit is directed at groups of knowledge workers that cooperatively search, annotate, interpret, and enrich large collections of heterogeneous documents from diverse locations. It is a generic set of components that can perform a variety of tasks such as learn new pattern recognition models, perform specialized search on resource collections, and store knowledge in a repository. W3C standards are used to make data accessible and manageable with Semantic Web technologies such as OWL, RDF(S), and SKOS. AIDA is also based on Lucene and Sesame. Most components are available as web services and are open source under an

Apache license. AIDA is composed of three main modules: Storage, Learning, and Search.

2.2.1 Storage – The Metadata Storage Module

AIDA includes components for the storage and processing of ontologies, vocabularies, and other structured metadata in the Storage module (see Annotation, Storage, and Ontology editing in Fig. 2.1). The main component is RepositoryWS, a service wrapper for Sesame[8] – an open source framework for storage, inferencing, and querying of RDF data on which most of this module's implementation is based [9]. ThesaurusRepositoryWS is an extension of RepositoryWS that provides convenient access methods for SKOS thesauri. The Sesame RDF repository offers an HTTP interface and a Java API. In order to be able to integrate Sesame into workflows we created a SOAP service that gives access to the Sesame Java API. We accommodate for extensions to other RDF repositories, such as the HP Jena, Virtuoso, Allegrograph repositories, or future versions of Sesame, by implementing the Factory design pattern. This pattern will allow parallel implementations of the Repository service to coexist.

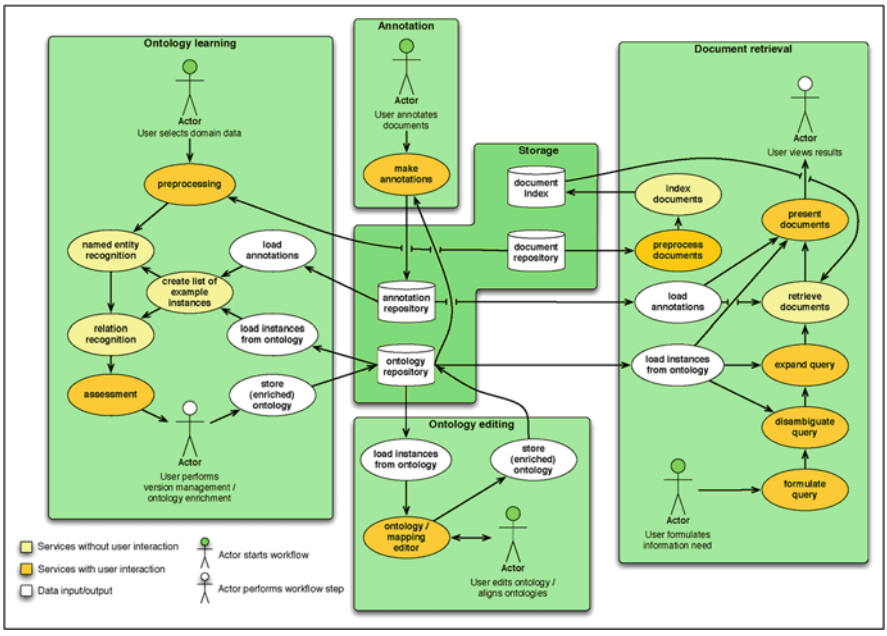


Fig. 2.1 AIDA components can be applied in a variety of activities

[8] Sesame and related RDF software is available from http://openrdf.org

RepositoryWS creates access to operations that enable the manipulation of an RDF repository, but does not contain any specific reasoning facilities for RDF-based knowledge representation languages such as OWL or FOAF. In order to support simplified access to domain vocabularies, we implemented a set of convenience methods for SKOS on top of RepositoryWS called ThesaurusRepositoryWS. SKOS is an RDF-based language for the representation of thesauri. ThesaurusRepositoryWS contains operations that enable querying for terms that represent a concept, their synonyms, broader, narrower, and related concepts, and mappings to concepts in other thesauri. Currently, the most common use of the thesaurus services is for browsing and searching vocabularies that have been stored in the repository. By starting at the top concepts (i.e., the "broadest concepts") of a vocabulary and progressively showing "narrower" concepts, an interactive hierarchical view of the vocabulary is provided in web browsers and the VBrowser application. Two example web service clients that make use of thesaurus operations have also been made: ThesaurusSearch for matching strings to concepts (i.e., searching for concepts in a thesaurus) and ThesaurusBrowser for looking up related concepts (i.e., navigating a thesaurus).

Most RDF manipulation will occur within workflows or applications that access RepositoryWS or ThesaurusRepositoryWS. Because most of our applications require user interaction, several examples of user interactions have been made available in AIDA clients such as HTML web forms, AJAX web applications, and a Firefox toolbar. The clients access RepositoryWS for querying RDF through the provided Java Servlets. An RDF web page demonstrates how to access web service clients from HTML forms. A combination of AJAX code and Java Servlets was used to create a web-based AIDA Thesaurus Browser. The AIDA Thesaurus Browser was used to create Recall samples for the Ontology Alignment Evaluation Initiative.[9] Another example can be found in the XUL Firefox extension for the annotation of web pages, that also access RepositoryWS through Java Servlets. Another example annotation client, similar to the Firefox extension, was implemented as an interactive web page that demonstrates auto-completion on labels of RDF Classes and Properties.

In HCLS IG work on federation of knowledge bases, we added several features to the Storage module. Our applications can now automatically detect and connect to a repository that is either Sesame, Virtuoso, or AllegroGraph. What was originally the Thesaurus Browser in our client user interface has now become the Repository Browser, because it can browse OWL hierarchies, as well as SKOS. Currently, autodetection also works to determine the contents of the repository: the SKOS specification from 2004 is attempted, followed by SKOS 2008, OWL classes, and a number of other specialized patterns. It is also possible to supply the entity types and relations as analogs to the SKOS:Concept and SKOS:narrower that were originally used in the ThesaurusRepositoryWS in functionality now called the *SKOS Lense*. This enables the user to customize the hierarchical browsing

[9] http://www.few.vu.nl/~wrvhage/oaei2007/food.html

to specialized types of RDF. The new functionality made it possible to easily access a wide variety of RDF from several different repository types by simply entering a URL to the repository host and choosing a repository or named graph.

The web services in Storage have recently been updated from the Sesame 1.2 Java API to the Sesame 2.0 Java API. Some of the new features that Sesame 2.0 provides, such as SPARQL support and named graphs, have been added to our web service API's and incorporated into our applications.

2.2.2 Learning – The Machine Learning Module

AIDA includes several components which enable information extraction from text data in the Learning module. These components are referred to as learning tools. The large community working on the information extraction task has already produced numerous data sets and tools to work with them. To be able to use existing solutions, we incorporated some of the models trained on the large corpora into the named entity recognition web service NERecognizerService. These models are provided by LingPipe [10] and range from the very general named entity recognition (detecting locations, person, and organization names) to the specific models in the biomedical field created to recognize protein names and other bio-entities. We specified several options for input/output, which give us an opportunity to work with either text data or the output of the search engine Lucene. The latter scenario is beneficial for a user who intends first to retrieve documents of his interest and then to zoom into pieces of text which are more specific. Output can be presented as a list of named entities or as the annotated sentences.

However, such solutions may not comply with the users' needs to detect named entities in domains other than the biomedical domain. To address this problem, we offer LearnModel web service whose aim is to produce a model given the annotated text data. A model is based on the contextual information and use learning methods provided by Weka [11] libraries. Once such a model is created, it can be used by the TestModel web service to annotate texts in the same domain. Splitting the entire process in two parts is useful from several perspectives. First of all, to annotate texts (i.e., to use TestModel), it is not necessary for a user to apply his own model. Given a large collection of already created models, he can compare them based on the 10-fold cross-validation performance. Another attractive option for creating models is to use sequential models, such as conditional random fields (CRFs), which have gained increasing popularity in the past few years. Although hidden Markov models (HMM) have often been used for labeling sequences, CRFs have an advantage over them because of their ability to relax the independence assumption by defining a conditional probability distribution over label sequences given an observation sequence. We used CRFs to detect named entities in several domains like acids of various lengths in the food informatics field or protein names in the biomedical field [12].

Named entity recognition constitutes only one subtask in information extraction. Relation extraction can be viewed as the logical next step after the named entity recognition is carried out [13]. This task can be decomposed into the detection of named entities, followed by the verification of a given relation among them. For example, given extracted protein names, it should be possible to infer whether there is any interaction between two proteins. This task is accomplished by the RelationLearner web service. It uses an annotated corpus of relations to induce a model, which consequently can be applied to the test data with already detected named entities. The RelationLearner focuses on extraction of binary relations given the sentential context. Its output is a list of the named entities pairs, where the given relation holds.

The other relevant area for information extraction is detection of the collocations (or n-grams in the broader sense). This functionality is provided by the CollocationService which, given a folder with text documents, outputs the *n*-grams of the desired frequency and length.

2.2.3 Search – The Information Retrieval Module

AIDA provides components which enable the indexing of text documents in various formats, as well as the subsequent retrieval given a query, similar to popular search engines such as Google, Yahoo!, or PubMed. The Indexer and Search components are both built upon Apache Lucene, version 2.1.0 (http://lucene.apache.org). We have chosen to extend this particular open source software suite for our information retrieval components because of the long-standing history, as well as very active user/developer community. This also means that indexes or other systems based on Lucene can easily be integrated with AIDA.

Before any document set can be made searchable, it needs to be processed – a procedure known as indexing. AIDA's Indexer component takes care of the preprocessing (the conversion, tokenization, and possibly normalization) of the text of each document as well as the subsequent index generation. It is flexible and can be easily configured through a configuration file. For example, different fields can be extracted from each document type, such as title, document name, authors, or the entire contents.

The currently supported document encodings are Microsoft Word, Portable Document Format (PDF), MedLine, and plain text. The so-called DocumentHandlers which handle the actual conversion of each source file are loaded at runtime, so a handler for any other proprietary document encoding can be created and used instantly. Because Lucene is used as a basis, there is a plethora of options and/or languages available for stemming, tokenization, normalization, or stop word removal which may all be set on a per-field, per-document type, or per-index basis using the configuration file.

An index can currently be constructed using either the command-line, a SOAP web service (with the limitation of 1 document per call), or using the Taverna

plugin. Once an index is created it can be searched through, using the AIDA Search component. There are three distinct ways of interacting with an index: (i) through a SOAP web service, (ii) using AJAX tools (based on JSON objects), or (iii) through a web interface (made using the ExtJS framework[10]). All methods use and "understand" Lucene's query syntax.

The Search SOAP web service (org.vle.aid.lucene.SearcherWS) can handle two kinds of queries, which either search through a single (document) field (called "search") or through multiple fields at the same time (called "searchMFquery"). The operation named "searchContent" is a convenience method which searches the content field by default, thus eliminating one parameter.

The Search web interface uses the JSON search operation, called "searchJason." Since AJAX cannot handle SOAP messages, there is a servlet which bridges the gap between the SOAP web service and the AJAX/JSON: org.vle.aid.client.jason. Additionally, the web interface can display a thesaurus, loaded through AIDA's Storage components. This thesaurus "view" may then be used to look up terms, synonyms, broader and narrower terms, and to perform interactive query expansion. The generality of using JSON for searching is clearly demonstrated by the fact that the output of this servlet can be used directly in Web 2.0 tools, such as Yahoo! Pipes or MIT's Exhibit.

2.3 Applications of Adaptive Information Disclosure

2.3.1 Food Informatics – Adaptive Information Disclosure Collaboration

The collaboration between the Adaptive Information Disclosure subprogram (middleware layer) and the Food Science subprogram (application layer) began early in the VL-e project in 2004, with regular meetings. AID had members from two universities in Amsterdam (University of Amsterdam and the Free University) and TNO, a national research institute. The Food Science partners include both industry and academia, with members at Wageningen UR, TI Food and Nutrition (TIFN), TNO Quality of Life, Unilever, and Friesland Foods. Member organizations of the collaboration were located in diverse remote locations several hours of travel apart, so there was substantial motivation to find ways of collaborating remotely. Of course, the use of e-mail served to enhance initial communications, with an eventual mailing list and archive devoted to the collaboration. Initial efforts focused on defining the possibilities and applications of the machine learning, information retrieval, and Semantic Web to Food Science.

Multi-disciplinary collaboration is a formidable challenge, even when the collaboration is between seemingly closely related disciplines such as the groups of machine learning, information retrieval, and Semantic Web that are represented

[10] http://www.extjs.com/

within AID. Different terminologies and approaches to problem solving and software implementation lead to an intensive process of checking intentions and agreements. Of course, this process is not unlike the process that every software engineer goes through when establishing software deliverables. There is not yet an established "mainstream" terminology for verbal discourse about knowledge, despite ample history provided by the fields of philosophy, logic, and artificial intelligence. So, for example, a knowledge engineer might use the word *metadata* to refer to semantic as well as syntactic-type information about data, whereas a database engineer will typically understand *metadata* to refer to table structure and syntactic-type information.

It is generally difficult to match the problems and tasks of an application domain with the capabilities provided by a new technology. Some of the difficulty is due to the knowledge gap between middleware developers and the users from a particular application domain (we will call it the *middleware gap*). The users find it difficult to understand what the possible applications are of a piece of middleware. On the other hand, the middleware developers do not usually know enough about the target domain to explain the possible applications of their middleware to the problems of that domain. The only way to bridge the middleware gap is either for the domain experts to become experts in the middleware or for the middleware developers to become knowledgeable about the application domain. Although no one is obligated to bridge the gap, it must be done in order to arrive at practical solutions.

With the goal of the collaboration being to use Semantic Web to effectively disclose information within and between the collaborating organizations, several areas of focus were identified. Establishing the vocabularies of a given research domain is an essential first step in defining the terms of discourse and interaction. In many domains, including biomedical research, a number of vocabularies had been established before Semantic Web standards reached Recommended status but are now available in the SKOS format. In order to take advantage of such vocabularies, we provide centralized access to important vocabularies such as AGROVOC, NALT, MESH, and GEMET via the AIDA thesaurus web services. However, although some existing agricultural vocabularies were related to Food Science, the development of structured vocabularies and ontologies specific to Food Science tasks would also be necessary. Document collections and query logs were also identified that could serve as potential sources for the extraction of customized vocabularies. Related work led to a knowledge acquisition method called Rapid Ontology Construction (ROC) [14] as well as an ontology of units and measures.[11]

A few areas of Food Science research serve as application use cases for the technology developed in the collaboration. One research area is centered on the study of bitterness and the conditions (ingredients, processing, etc.) in which it arises. Bitter perception affects the enjoyment of many foods and can even serve as an indicator of toxicity. A database of bitter compounds was developed at Unilever called BitterBase and web services were developed that could use information about

[11] http://www.atoapps.nl/foodinformatics/NewsItem.asp?NID=18

a food component or molecule to predict its bitterness [15, 16]. The BitterBase web services were combined in a Taverna workflow to create a demonstration. The BitterBase web services can eventually serve as a link to knowledge about chemical compounds in other applications. Another area of interest is Food Safety. Incidents where toxic chemicals have entered the food-supply chain have resulted in huge losses for the food industry and reduced consumer confidence in food. The Early Warning System of TNO is being developed for the early detection of food safety risks in the agro-food supply chain. Having made an ontology for food, our Food Science partners were able to browse the OWL class hierarchy of the ontology after we converted it to SKOS and created a web client that accessed it via the AIDA thesaurus web services.

In interactive AIDA applications, the thesaurus services are employed to create a hierarchical browser of the terms in a structured vocabulary, allowing the user to navigate from the most general concepts down to the most specific with mouse clicks. Search is also available to find concepts whose labels match string patterns. These interactions make it possible to navigate large vocabularies, as well as the concepts of a custom-built food ontology. In the case of the ontology, rdfs:subClassOf in the ontology is mapped to skos:narrowMatch in a SKOS version of the subclass hierarchy. Such a mapping can be used to convert OWL to SKOS using a SeRQL Construct query.

An important application of vocabularies is query expansion and refinement. In principle, adding a "narrower" term to a query does not change the intention of the query. In fact, it should improve recall when we are searching for the presence of the query terms in the documents such as is done in Lucene. SKOS allows us to look up terms that are "narrower" than a given term programmatically, so that we can automatically perform query expansion based on those terms when we find our query terms in the vocabulary. For example, if our query contains the word "bread," and we find the narrower terms "ciabatta" and "bread crumbs," we can automatically add both terms to the query. Our ThesaurusRepositoryWS offers this capability in the method "getNarrowerTerms." This functionality was incorporated into the research management system called Tiffany, developed in a collaboration between TIFN and Wageningen UR and used at TIFN. The AIDA web interface that is used in TNO's Early Warning System incorporates narrower terms with a Query Builder and search of a Lucene index. The narrower terms are used in a new way: the drag of a concept to the Query Builder recursively adds all narrower terms "below" it, representing the concept with as many terms as possible within the given vocabulary. In the example shown in Fig. 2.2, the resulting search is for a long list of compound names in the index.

2.3.2 A Metadata Management Approach to fMRI Data

The data from medical imaging experiments brings with it a fundamental problem: the system in which the data is stored can make it difficult or impossible to search

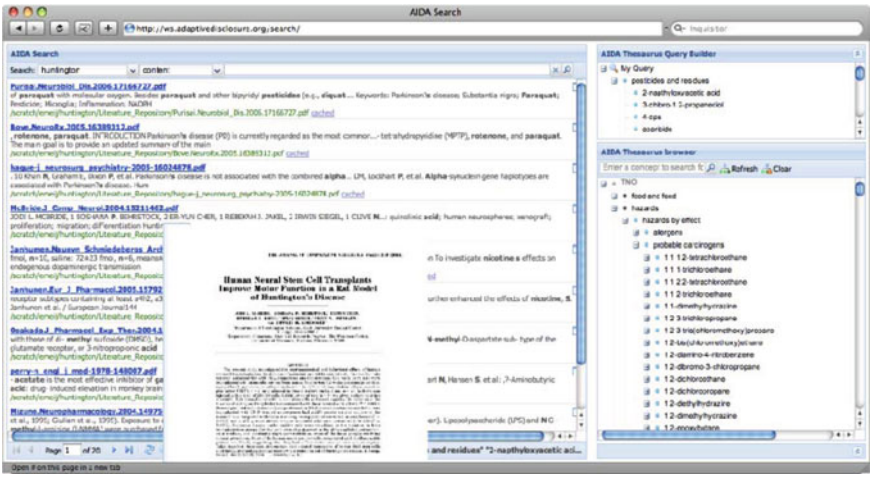


Fig. 2.2 Concept search (*upper right*) for "pesticides and residues" (dragged from *lower right*) using SKOS relations for query expansion to search Huntington's disease corpus (with open Document Viewer for one result document)

for data with a particular set of attributes. However, many types of analysis require this type of search. In medical imaging, different data sets can result from different acquisition protocols, subjects, studies, etc. Processed data can also result from different image analysis workflows and typically include intermediate and final results obtained from different algorithms or parameter settings. In this section, we describe our approach to the management of functional Magnetic Resonance Imaging (fMRI) data using a metadata management plugin for the VBrowser.

Functional MRI (fMRI)[17] enables the non-invasive study of brain activation by acquiring images while the subject is performing some physical or cognitive activity in response to controlled stimulation. The raw data consists of functional and anatomical images, stimulus data and other signals, which are submitted to a complex analysis workflow to compute Brain Activation Maps (BAMs). An ongoing study at the AMC has adopted grid technology to investigate the effect of acquisition and analysis parameters on the resulting BAMs (Virtual Lab for fMRI[12]). This study generates a large amount of data that needs to be properly annotated to facilitate retrieval for result interpretation, preparation of publications, and sharing with other researchers.

The Virtual Resource Browser (VBrowser) is a user interface that provides access to data resources on the grid [18]. The adoption of grid technology makes the VBrowser an attractive interface choice because it provides support for basic tasks such as direct data manipulation and transfer (view, delete, move, etc.) using grid protocols such as SDSC Storage Resource Broker, gridFTP, and gLite Logical

[12] www.science.uva.nl/~silvia/vlfmri

File Catalog. Moreover, the VBrowser is extensible via plugins that implement, for example, access to grid and web services. The VBrowser is the primary front-end to the Virtual Lab for fMRI.

A set of web services from the AIDA Toolkit were incorporated as a VBrowser plugin that makes it possible to annotate and retrieve grid resources with RDF. For the fMRI application, we have created a metadata schema in OWL that is used for annotation. This allows users and programs to annotate a given fMRI image or experiment result with associated parameters and their values. Using another VBrowser plugin, the user can initiate and monitor experiments that perform large-scale fMRI analysis on the grid. In these experiments, a parameter sweep is performed across the values and the result obtained with each parameter combination is annotated with the corresponding values. At a later stage, users can retrieve results based on concepts available as knowledge resources in the AIDA plugin for VBrowser. The adoption of concepts associated with the parameters of interest in the study enhances usability by enabling the user to express queries in more familiar terms. The query results are presented as a list that can be browsed directly on the VBrowser. Query results are stored as permanent resources that can be reused, refined, and shared among researchers involved in related studies. Besides the ability to query, our metadata approach also allows for the addition of concepts and terminologies from other domains, making it possible to select images based on concepts that are more directly related to the subject of study (e.g., area of brain, type of neuron, type of activity, disease) as well as image features and image quality (Fig. 2.3).

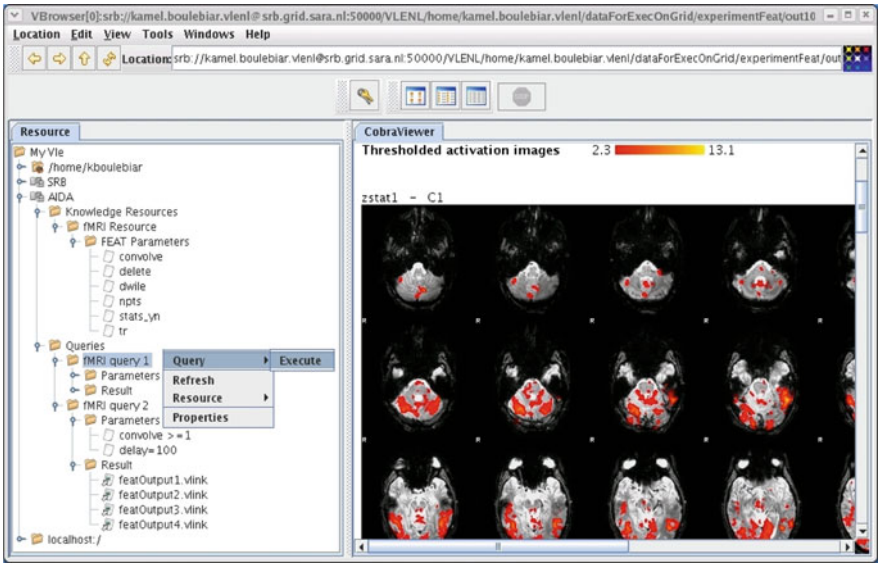


Fig. 2.3 The *right-hand* pane shows the view of a query result. The *left-hand* pane shows fMRI analysis parameters as knowledge resources that can be used in queries. Queries can be edited, saved, and executed from the Queries folder of the AIDA resource

2.3.3 Semantic Disclosure in Support of Biological Experimentation

In this section we treat two related biological cases for semantic disclosure. In the first we demonstrate the disclosure of human genome data for cases when we would like to compare different data sets to test a biological hypothesis. In the second we address the case of hypothesis formation itself that has become a formidable task for biologists. In both cases Semantic Web languages and tools help to disclose data and knowledge for use in computational experiments.

2.3.3.1 Application Case 1: Semantic Disclosure of Human Genome Data

Over the last decades it has become increasingly clear that the activity of one gene is regulated within the context of large networks of activities in the cell, including the activities of many other genes. Instead of investigating genes one by one, we are now able to perform genome-wide studies as a result of the Human Genome Project and many of its followers that provided whole genome sequences and genome-associated data such as gene expression profiles or binding locations of various DNA-binding proteins. Typically, data is stored by a data provider in a relational database and users access this data either by downloading it from the provider's web site or by interacting with the provider's web user interface. A typical example that we will use is the UCSC genome browser ([19]; http://genome.ucsc.edu), one of the largest resources of genome data. One way to analyse this data is by comparing selections of data through the visual interface of the provider. The USCS genome browser shows genome data as stacked "tracks," where a track could be a gene expression profile along a chromosome. This type of "visual integration" is of course no longer appropriate when we want to perform a more complicated analysis of several data sets, especially if we want to be able to repeat and rigorously evaluate the analysis. That requires a computational approach. The traditional approach is to download the data sets to a local database and query the tables locally through SQL queries or via specific scripts. In most of these cases, the relational models used by either the provider or the user are knowledge meager. For instance, UCSC tables typically consist of rows of at least four columns, one for the chromosome number, one for a start position on the chromosome, one for an end position, and one for a value. Any meaning beyond the name of the columns is not directly linked with the data. If we want to find out more about what the data means, its biological context or how it was created and by whom, we will have to follow the hyperlinks on the UCSC web site and read the descriptions and the papers that the web pages refer to. Consequently, knowledge that is relevant for a particular data integration experiment is known by the researcher, but not by the computer. We will not be able to use it directly for computational data integration. In this section we show how we can link data to one's own semantic model and how we can use this to achieve semantic data integration. We demonstrate how this allows us to address different data sets through our own concepts and terms. In other words, we disclose the semantics for our data integration experiments.

Integrating Data for a Specific Hypothesis: DNA-Binding Sites of Transcription Factors and Histones

An intrinsic aspect of our approach is the focus on specific hypotheses within a biological research context. Within the broader context of investigating the relationship between how DNA is structured in the cell and transcriptional activity of genes, the question of how histone-binding sites relate to transcription factor-binding sites is of interest (Fig. 2.4). Histones are specific types of proteins that bind DNA and as such are central to packaging long DNA molecules in the nucleus of a cell. They undergo specific chemical modifications that influence their position and binding affinity for DNA, which can have an effect on transcriptional activity. "Transcription factors" are also proteins that can bind DNA, but they typically influence gene expression directly by binding specific sequences near specific genes. Many of these sequences have been identified and localized on human DNA. The interplay between histones and transcription factors is of interest and therefore we would like to be able to query DNA positions of both types of proteins. As mentioned above, we can find the appropriate data in the UCSC genome browser; the ENCODE project provided data for histone-binding tracks and transcription factor-binding tracks [20]. We can view these tracks together for visual inspection, but for any kind of genome-wide analysis this does not suffice. Below we show how we enable ourselves to perform these studies computationally by providing access to these data sets via our own semantic model. The model contains the biological concepts and relationships that we consider relevant for our biological hypothesis, where we assume a correlation exists between the binding of specific modified histones and specific transcription factors (Fig. 2.4).

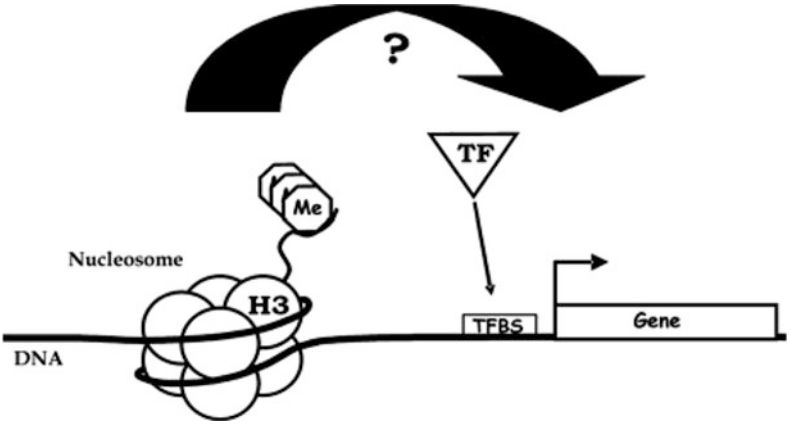


Fig. 2.4 Cartoon model representing a biological hypothesis about the relationship between positions of histones and transcription factors

"myModel" for Experimental Scientists

An important aspect of experiment biology that we would like to take into account is that disagreement among peers is a key factor for scientific progress. Ontologies often aspire to capture common agreement, but semantic models can also be used to capture the view of an individual scientist or research group. Taking this into account, we can follow these steps to integrate data:

1. Create or extend "myModel".
2. Semantic disclosure: link data sets to myModel (Fig. 2.5).
3. Perform integrative analyses using elements from myModel.
4. For new experiments return to Step 1.

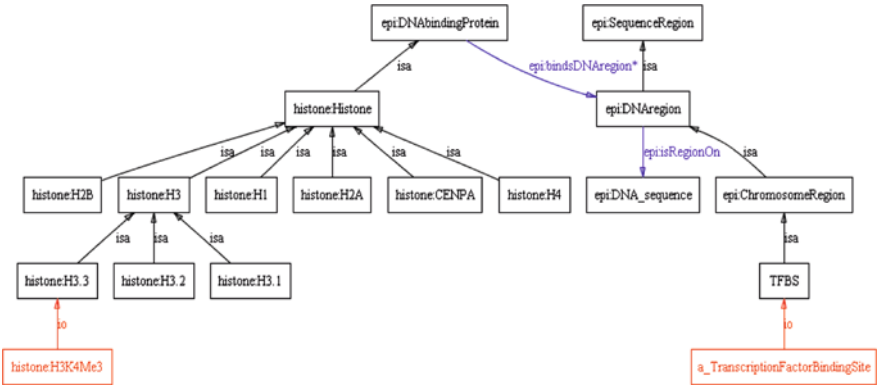


Fig. 2.5 Principle of semantic data integration. The rectangles represent classes in our OWL model: histone classes on the left and transcription factor classes on the right. The bottom two rectangles refer to data items from two different datasets (represented here as OWL instances). The histone model and transcription factor model are linked by biological properties. Therefore, linking data about histones and data about transcription factors to this model creates a biologically meaningful link between the two data sets

In some cases it may be worthwhile to use existing (de facto) standard controlled vocabularies such as the Gene Ontology, but here we follow the scenario where we require a personal model and the ability to make full use of OWL modeling possibilities. At a later stage we can map terms from more comprehensive ontologies to our purpose-built model, if needed. Initially, myModel need not be larger than the next biological analysis requires. We can exploit the extendibility of OWL for each new experiment.

Technical: From Their Data to my Model

Once we have created a semantic model in OWL, how do we link it to the data sets that we wish to query (step 2 in the previous paragraph)? In our example case we have tables of chromosomal binding locations of histones and transcription factors.

We also have the relational schema for these tables from UCSC. However, for linking with our OWL model we require data to be available in RDF or at least have an RDF interface. If the provider does not provide this, then one possible procedure is as follows:

1. Identify the data sets required for the experiment
2. Convert the provider's table schemas or column headers into a RDF Schema (theirDataModel).
3. Convert the data to RDF by linking the values in the data sets to the concepts in theirDataModel
4. Create semantic links between theirDataModel and myModel.

We expect that increasingly more providers will follow the example of UniProt (the main protein data provider) and provide a RDF interface to their data. This would allow us to skip steps 2 and 3. Otherwise, a conversion or some kind of mapping to RDF is inevitable [21–26]. An ideal situation would be if data producers (the wet laboratories), data providers, and data users would each provide their semantic models and their relations to the data (Fig. 2.6).

It is generally advisable to create models for representing the data (preserving the data supplier's naming scheme) and models to represent biological knowledge, with an explicit mapping to link them. We could have imported the raw data directly into myModel, but this would be less flexible and less robust. For instance, changes to our biological models could require an entire new conversion process for any

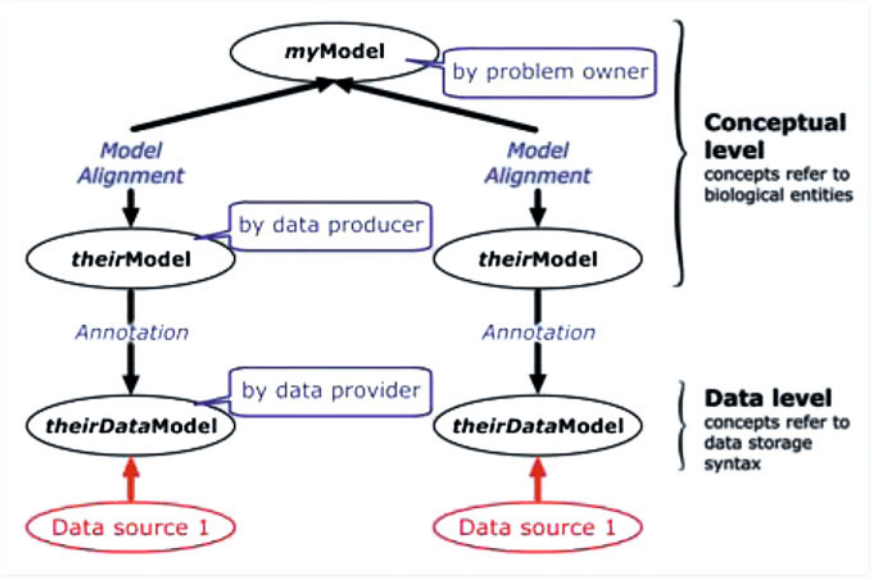


Fig. 2.6 Principle of semantic disclosure in an ideal world. All parties involved provide a semantic model conform their role with respect to the data. Data integration is achieved by aligning semantic models linked to different data sets

data that is affected, whereas we only need to change the mapping in the case of separated models.

Analyzing Semantically Integrated Data

By semantically disclosing data we achieve a situation where we can address separate data sets via biological concepts and relations in our own model. We conclude by an example query (in "pseudo-SeRQL") that retrieves those regions of DNA where both histones and transcription factors bind (see [22–24] for other examples). The "domain of comparison" in this experiment is "ChromosomeRegion." Note that the query does not need to contain direct references to the UCSC data sets.

```
SELECT histone, transcriptionfactor, chrom1, tStart1, tEnd1
FROM {histone_region} myModel:Chromosome_identifier {chrom1};
      myModel:hasStartLocation {tStart1};
      myModel:hasEndLocation {tEnd1};
      myModelExperiment:hasMeasurementValue {score1},
    {histone} myModel:bindsDNAregion {histone_region};
      rdfs:type {myModel:Histone} rdfs:type {owl:Class},
    {tf_region} myModel:Chromosome_identifier {chrom2};
      myModel:hasStartLocation {tStart2};
      myModel:hasEndLocation {tEnd2};
      myModelExperiment:hasMeasurementValue {score2},
    {transcriptionfactor} myModel:bindsDNAregion {tf_region};
      rdfs:type {myModel:TranscriptionFactor} rdfs:type
    {owl:Class},
WHERE chrom1 = chrom2 AND (tStart1 <= tEnd2 AND tEnd1 >=
tStart2)
```

The above query is sometimes called a *Stand-off Join* or *Interval Join*, where the boundaries of two intervals are compared in order to see if there is overlap. This join is frequently necessary when scanning for voice annotations (e.g., reviewer's comments) over a film in a multimedia database. We note that this type of query is challenging for any non-optimized database and certainly also for the semantic repositories we tested some years ago [25]. Performance was generally very poor for all the databases that we tried, including relational databases. However, the XML database MonetDB [27] which is optimized for precisely this type of stand-off join performed better by orders of magnitude. We conclude that considerable performance gains are to be expected considering that semantic repositories are still immature in this respect.

2.3.3.2 Application Case 2: Semantic Disclosure of Biological Knowledge Trapped in Literature

A common way to study intracellular mechanisms in biology is via cartoon models that represent a particular hypothesis. A typical example is a cartoon that represents

a hypothesis about the compaction of DNA, a key factor for sustained regulation of gene expression (top left in Fig. 2.8; see also [28]). Hypotheses can contain many different types of information: sequences, biophysical entities such as proteins and lipids, 3 D structural information, biochemical reactions. They are the basis for each experiment in the laboratory. Given that laboratory experiments are expensive in terms of money and effort, hypothesis generation is an area of interest for bioinformatics and e-Science. Forming a good hypothesis requires the integration of increasingly large amounts of resources. There are over a thousand public databases with experimentally derived data available to biologists and over 17 million biomedical publications are available via the prime knowledge resource for most medical and molecular biologists, Entrez PubMed.[13] PubMed gives access to the National Library of Medicine's public digital library MedLine and several other resources. It is increasingly challenging to ensure that all potentially relevant facts are considered while forming a hypothesis. Support for retrieving relevant information is therefore a general requirement. This leads to the question of how to disclose this information such that it becomes a resource for computer-aided hypothesis generation. Preferably, this process is under the control of a biologist as much as possible, considering that no one else has a better understanding of the end goal: a better biological hypothesis. This presents a problem, because automated information extraction is generally not the area of expertise of a biologist. In this section we show how an e-Science approach based on the application of (AIDA) Web Services, Workflow, and Semantic Web technology enables application scientists to exploit the expertise of scientists from various disciplines for building a machine readable knowledge base as a resource for hypothesis generation.

2.3.3.3 An e-Science Approach for Extracting Knowledge from Text

In order to demonstrate our approach, we will discuss an application in which we would like to extend a hypothesis about condensation and decondensation of chromatin. This is an important determinant of gene expression, because the effects can be sustained over generations of proliferating cells. In particular we would like to investigate putative relationships between the protein "Histone deacetylase 1" (HDAC1), involved in condensation, and other proteins. A traditional scenario would be to query PubMed and browse through the documents it retrieves (over 300 for the query "HDAC1 and Chromatin"). We obtain a "feel" for what is important and read a selection of papers for more in depth information. However, this selection would probably be biased. Extracting the proteins related to HDAC1 without human bias would be at least a highly laborious task. We also have to consider that subsequent experiments based on alternative hypotheses will require new searches that preferably extend our previously obtained information. Therefore, we would like to address this problem computationally.

[13] http://www.ncbi.nlm.nih.gov/pubmed/

Our objectives are to

1. Extract specific knowledge from literature, proteins in the case of our example
2. Examine *all* relevant papers or at least papers selected without subjective bias
3. Store the results in a structured way such that they fit our biological hypothesis and can be re-examined and extended
4. Enable biologists or bioinformaticians to design their own knowledge extraction "experiments"

In line with experimental science, we regard the whole knowledge extraction procedure as a "computational experiment", analogous to a wet laboratory experiment. Such an experiment requires an insightful and re-executable design of which the results are structured enough to allow us to retrace evidence. In the wet laboratory analogy we would use a laboratory journal for the latter.

We can achieve objectives 1–3 by implementing a basic text mining procedure [2, 3] as follows:

(1) Retrieve appropriate documents from Medline (information retrieval)
(2) Extract protein names from their abstracts (information extraction)
(3) Store the results for later inspection. The results have to be linked to our hypothetical model and we need to store the evidence that led to these results. Evidence (provenance) in this case is, for instance, the documents from which protein–protein relationships were derived and the computational resources that were used.

Taking another look at these basic steps we see the desire for a multidisciplinary approach. Step 1 is one of information retrieval, step 2 can be done using machine learning techniques, and for step 3 Semantic Web formats and tools can be used. Each of these steps relates to a distinct scientific discipline. Instead of one bioinformatician reinventing many wheels, an e-Science approach leverages expertise from disparate fields for an application. We achieve this when scientists in these fields produce Web Services as part of their activities and make them publicly available. We call this "collaboration by Web Services." In our case, PhD students in three research groups produced the Web Services and infrastructure that we need for our application (see Section 2.2).

Making use of a service-oriented approach also allows us to meet objective 4. AIDA Web Services and others can be strung together with a tool such as Taverna[14] [29] to form an executable workflow that performs the knowledge extraction procedure (Fig. 2.7). The workflow reflects the basic steps of the text mining procedure and its design can be stored and reused. For instance, we store our workflows on myExperiment.org,[15] a "web2.0" site for computational scientists to store and share

[14] http://www.mygrid.org.uk/tools/taverna/
[15] http://www.myExperiment.org

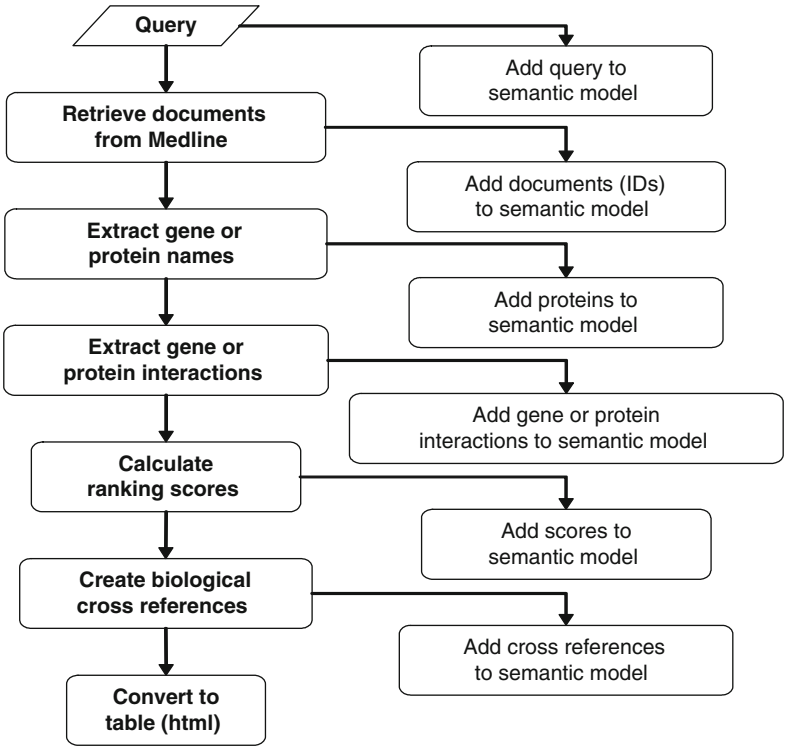


Fig. 2.7 Workflow for extracting proteins from literature (*left*) and store them in a knowledge base (*right*). We added steps to provide a likelihood score, cross-references to some popular biological databases, and tabulated results

(publish) computational artifacts such as workflows. The final step of the procedure, storing results, uses the Web Ontology Language (OWL) and the Resource Description Framework (RDF) to represent the knowledge that we want to extend with the text mining results. The next paragraph explains the knowledge modeling step in more detail.

Modeling for Biological Knowledge Extraction in OWL

Our general approach toward modeling hypotheses for computational experiments is to start with a "proto-ontology" that represents a minimal amount of knowledge appropriate to the problem at hand. In our example case the model should represent at least proteins and artifacts related to the experiment itself to enable us to express, for instance, that a protein was discovered in a particular document from Medline. The purpose of our workflow is to extract knowledge from text and populate this model with individual knowledge instances (e.g., a particular protein). We should consider that our observations from text mining are just pieces of text until

we have interpreted them and converted them into a machine readable form accordingly. For instance, we will interpret the term "p53" found in a particular abstract as an instance with label "p53" of the class "Protein," and its collocation with the term "HDAC1" as a (putative) biological relationship with the protein labeled "HDAC1." Obviously, "collocation" in text does not necessarily mean collocation in the biological sense. To prevent conflation of the biological view and observational views we create four distinct OWL models and one to map between these models (Fig. 2.8):

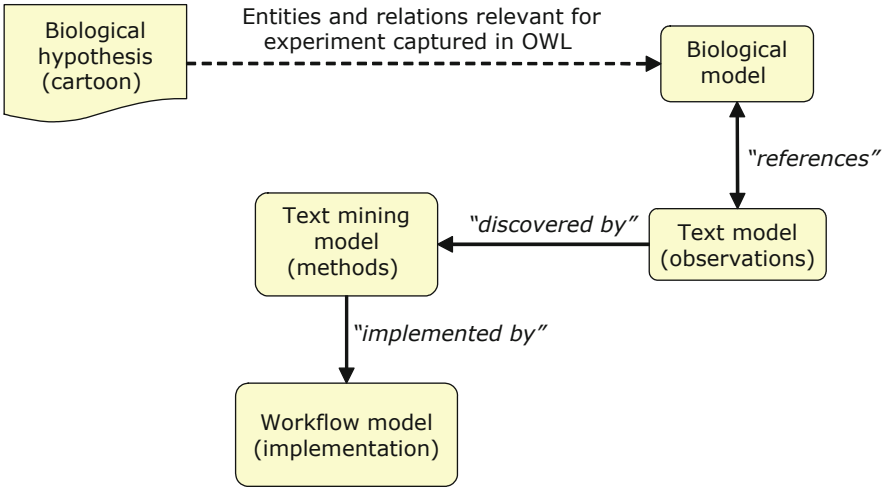


Fig. 2.8 Overview of models and their interrelationships. *Arrows* represent relationships between instances of classes between the models. These relationships are defined in a separate mapping model

1. *Biological model* The biological model is the primary model representing our biological hypothesis. It contains classes such as "Protein," "Interaction" and "Biological model." The model is not extensive, because part of our approach is that we do not define more classes than are necessary for our experiment. OWL allows us to extend the model later as needed by new experiments. We follow the concept of the biologist's cartoon model in that we do not necessarily try to represent real entities or relationships, but hypothetical models of them. Instances in this model are interpretations of certain observations, in our case of text mining results. The evidence for these interpretations is important, but it is not explicitly within the scope of this model.
2. *Text model* The text model contains classes such as "Document," "term," and "interaction assertion." Instances are the concrete results of the knowledge extraction procedure. We can directly inspect documents or pieces of text, in contrast to instances of the biological model such as proteins or DNA. Creating an instance in the document model leads to creating an instance in the biological model based on the assumption that if a protein name is found collocated

with another protein name we assume that the referred-to proteins participate in a biologically meaningful relationship.

3. *Text mining model* The text mining model represents the knowledge extraction process itself. It contains classes for information retrieval, information extraction, and the text mining process as a whole. In principle, these processes could be implemented in different ways. Therefore we created a separate model, in our case a workflow model. These models are linked by "implementation" relationships.
4. *Workflow model* The workflow model represents the computational artifacts that are used to implement the text mining procedure. Example instances are (references to) the AIDA Web Services and runs of these services. Following the properties of these instances we can retrace a particular run of the workflow.
5. *Mapping model* While we have a clear framework for representing our biological hypothesis, text, text mining, and workflow, we also need a way to relate the instances in these models. Therefore we created an additional mapping model that defines the reference properties between the models.

In summary, we have created proto-ontologies that separate the different views associated with a text mining experiment. We can create instances in these models and the relationships between the instances in these views (Fig. 2.9). This allows us to trace the experimental evidence for creating the instances in the biological model.

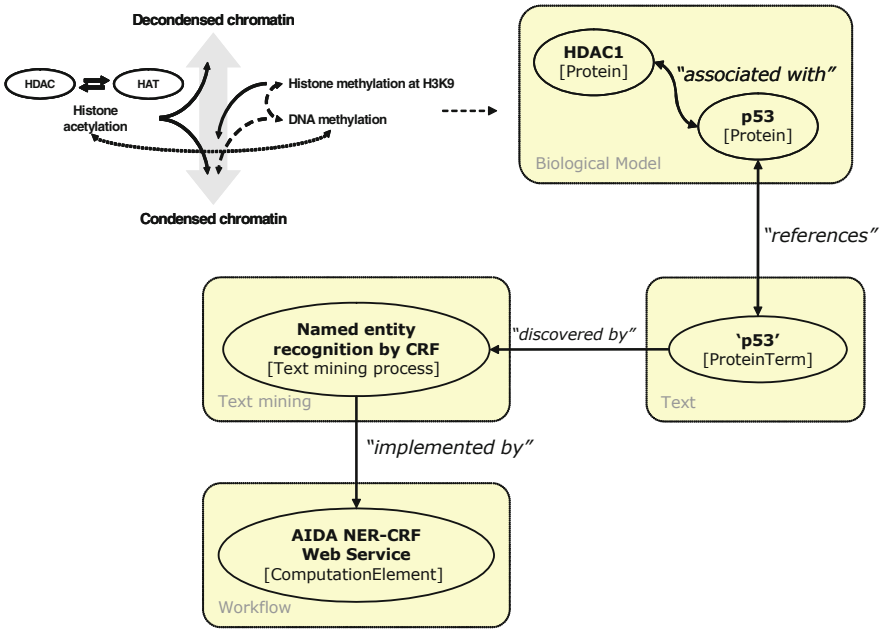


Fig. 2.9 Examples of instances and their relationships between the views associated with a text mining experiment

In our case of text mining, evidence is modeled by the document, text mining, and workflow models. A different type of computational experiment may require other models to represent evidence and new mappings.

We note at this point that in our example case the distinction between proteins and genes presents a problem. They are biologically distinct, but are typically referred to by the same name. There are simple typographic rules to distinguish between gene and protein names (e.g., *ftsQ* versus FtsQ), but these are not always adhered to, are lost in digital copies, or are overlooked by text miners. Therefore, text mining does not make this distinction generally. In our case, we chose to map the text mining results, instances of protein (or gene) names to instances of proteins by default. Alternatively, we could have defined a (biologically awkward) class "gene or protein" in the biological model and map protein (or gene) names in the text model to instances of that class.

A Repository for Storing and Retrieving Biological Knowledge

For our knowledge extraction experiment, we now have a workflow and proto-ontologies for structuring the results of the workflow. For storing this knowledge we use Sesame, a freely available RDF repository. We can add the proto-ontologies and let the workflow populate these ontologies with instances from the text mining procedure (see right side of Fig. 2.7). One of the conveniences of this approach is that by the relatively straightforward action of adding an instance with certain properties, the instance becomes linked with any additional knowledge that was previously added to the repository. For instance, when we add NF-KappaB to the repository and its relationship with a hypothesis about HDAC1, we find that NF-KappaB is also related to a hypothesis about nutrients and chromatin that was used in a previous experiment. Another practical convenience in comparison to relational databases is that referential integrity or preventing redundancy is largely handled by Sesame's built-in RDFS reasoner.[16] With AIDA services we can query and alter the content of a Sesame repository not only from within a workflow but also in a client such as VBrowser, a general purpose resource browser that has been extended by an AIDA plugin. In addition, we can manipulate semantic content by using the Sesame workbench user interface or the Sesame API.

Finally, OWL data can be used over the Internet to create a Semantic Web, because each node and edge of the underlying RDF are referred to by a Universal Resources Identifier (URI). This enables exploitation of powerful features, such as virtual integration of distributed models and data. However, to make OWL models truly part of the Semantic Web, we have to make sure that the URIs resolve properly. A simple way to do this is to ensure that the models are also stored as OWL files on a publicly accessible URL that corresponds to the base of the URIs used in the semantic models, in our case http://rdf.adaptivedisclosure.org/. It is also possible

[16] Sesame supports RDF reasoning for RDF-Schema repositories, not for RDF repositories.

to configure Virtuoso[17] to expose repository contents to URL access in the Linked Open Data tradition. Transparent access to the content of semantic repositories by means of URLs is currently subject to active research and development.

Results

The result of running the workflow is a knowledge base filled with instances of biological concepts, relationships between those instances, and links to instances that can tell us why the instances were created. The instances were classified according to our own proto-ontologies. We can examine the results in search of unexpected findings or we can trail the evidence for certain findings, for instance, by examining the documents in which some protein name was found. An interesting possibility is to explore relationships between the results of one or more computational experiments that added knowledge to the knowledge base. There are a number of ways to explore the knowledge base. We can load the models and instances in Protégé to use its browsing and reasoning features or we can use RDF query languages such as SeRQL and SPARQL. We can also access the models and new instances with our own web interface or Taverna plugin (see Fig. 2.10). It is also possible to perform

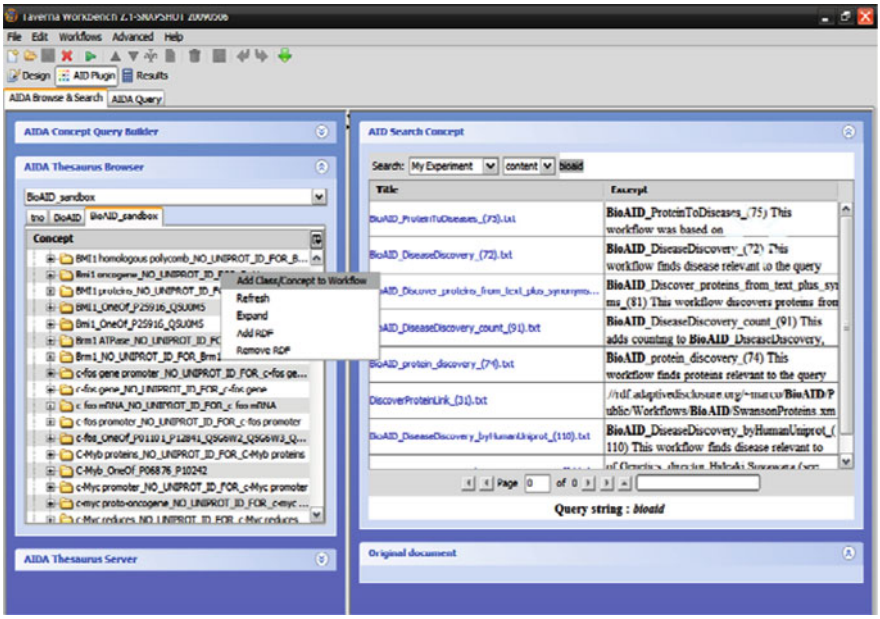


Fig. 2.10 The AIDA Taverna plugin makes it possible to browse and search the results of the workflow without leaving Taverna

[17] http://virtuoso.openlinksw.com/

another computational experiment by means of a workflow using the knowledge base. To illustrate the principle we provide three examples of RDF queries[18]:

1. This query retrieves instances of biological hypothesis models and their partial representation by a user's search query. The query tries to match "{node} *edge* {node}" patterns in the RDF graph. The prefixes refer to our proto-ontologies and standard models such as owl: and rdf:

```
SELECT model, label(query)
FROM {model} rdf:type {bio:BiologicalModel} rdf:type {owl:Class},
{representation} map:partially_represents {model},
{representation} meth:has_query {query}
```

Output:

Instance of biological model	query partially representing the model
http://rdf.adaptivedisclosure.org/owl/BioAID/myModel/Enriched-ontology/BioAID_Instances.owl#BioModel_HDAC1_AND_chromatin	"HDAC1 AND chromatin"

2. The following query retrieves proteins that are shared between two subsequent runs of the workflow with different biological models (hypotheses) as input. We consider an input query of a workflow as a (partial) representation of the hypothesis.

```
SELECT label(comment), label(query1), label(query2)
FROM {protein_instance} rdf:type {bio:Protein} rdf:type {owl:Class},
{protein_instance} rdfs:comment {comment};
       bio:isModelComponentOf {model1};
       bio:isModelComponentOf {model2},
{representation1} map:partially_represents {model1};
       meth:has_query {query1},
{representation2} map:partially_represents {model2};
       meth:has_query {query2}
WHERE model1 = inst:BioModel_HDAC1_AND_chromatin AND
   model1 != model2
```

[18]The examples here are a simplified version of SeRQL; for complete SeRQL examples including namespaces, see http://www.adaptivedisclosure.org/aida/workflows/bioaid-serql-query-examples

```
Output:
```

Protein	Query for model 1	Query for model 2
"protein referred to by as NF-kappaB and UniProt ID: P19838"	"HDAC1 AND chromatin"	"(Nutrician OR food) AND (chromatin OR epigenetics) AND (protein OR proteins)"
"protein referred to by as p21 and UniProt ID: P38936"	"HDAC1 AND chromatin"	"(Nutrician OR food) AND (chromatin OR epigenetics) AND (protein OR proteins)"
"protein referred to by as Bax and UniProt ID: P97436"	"HDAC1 AND chromatin"	"(Nutrician OR food) AND (chromatin OR epigenetics) AND (protein OR proteins)"

3. Finally, a query that retrieves a "trail to evidence" for a protein instance. It retrieves the process by which the name of the protein was found, the service by which the process was implemented, and its creator, the document from MedLine, that is the input for the service and contains the discovered protein name and the time when the service was run. In this case we depict the query as a graph emphasizing that RDF queries are in principle graph patterns that match patterns in the knowledge base (Fig. 2.11).

Overall, using the December 2008 version of our MedLine index the workflow created 257 protein instances linked to our biological model of HDAC1 and chromatin. They were discovered through 489 protein terms found in 276 documents and we could recover by what process, web service, and workflow these were discovered, and when. The discrepancy between the number of protein instances and protein names is because proteins are referred to by various synonyms. As our knowledge base grows with each experiment (not necessarily text mining), we can perform increasingly interesting queries in search of novel relations with respect to our nascent hypothesis. We can store these queries and share them as "canned queries". We are curious to see which queries will turn out to be biologically most revealing.

Semantic Data Integration by Knowledge Extraction Workflows

The text mining workflow above, although created for building a knowledge base to support hypothesis generation, can also be seen as a data integration or data annotation workflow. If we consider documents contained in MedLine as data elements, then we have annotated these data elements by linking them to proteins and several semantic relationships in our proto-ontologies. The semantic annotation is provided by mining the text that is associated with the data. We can now query these data elements via our own concepts and instances (which we may have linked to de facto standard ontologies for extra points of reference), as we could after semantically annotating human genome data from the UCSC genome browser (see elsewhere in this chapter and [25, 26]). One of the advantages of combining workflow and

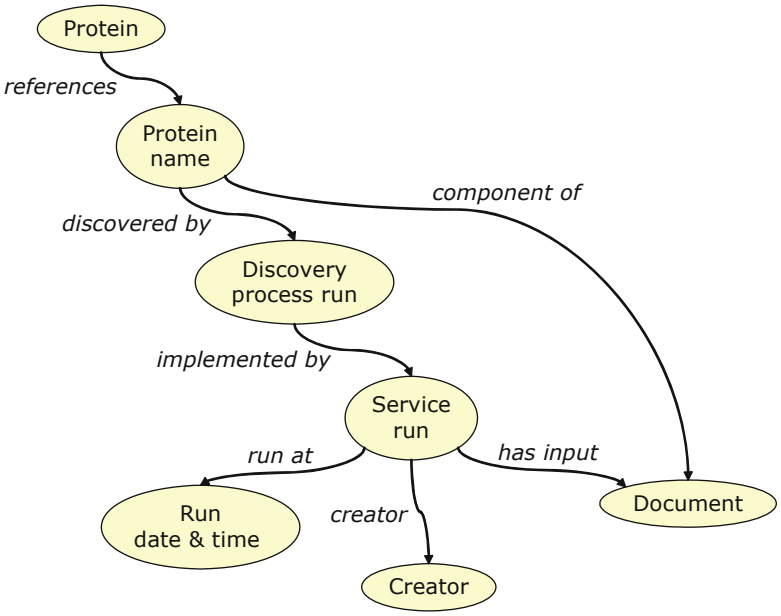


Fig. 2.11 Graph representing a RDF query to retrieve the evidence for a biological instance

Output for Protein identified by "P19838".
Protein: "Protein referred to by as NF-kappaB and UniProt ID: P19838"
Discovered by: "Named entity recognition trained by conditional random fields (CRF) on protein names"
Implemented by: "AIDA CRF Named Entity Recognition service"
Created by: "Sophia Katrenko (University of Amsterdam)"
Input/Component container: Document with PubMed ID 17540846[19]
Timestamp: "2008-11-18T03:29:30+01:00"

Semantic Web technologies in this way is that we can dynamically create a semantic "data warehouse" according to our own needs and wishes. This pattern could be applied whenever data is coupled to some form of free text.

2.3.3.4 Conclusion

In this section we have explored mechanisms for semantic disclosure of human genome data and knowledge enclosed in literature, both in the context of supporting hypothesis-driven experimentation. In the first part of this section we could demonstrate the principle of being able to query (experiment with) data in terms of our own semantic model. Perhaps more importantly, it showed an elegant way to integrate different data sets using Semantic Web formats and tools. Extracting knowledge

[19] http://www.ncbi.nlm.nih.gov/sites/entrez?cmd=Retrieve&db=PubMed&list_uids=17540846

from literature is a more challenging task requiring expertise from different fields of science. We showed that by using a workflow we can combine the expertise of several scientists for disclosing information trapped in literature and created a biological knowledge base that can be explored by semantic queries in search of biological hypotheses. The pattern of storing (interpretations of) results from a workflow into a knowledge base can be used repeatedly to build an increasingly rich resource for elucidating biological phenomena and may also be applied for automated data integration

2.4 Discussion

A number of technologies are implemented in AIDA, each with particular uses and advantages. For example, Sesame RDF repositories enable storage and retrieval of knowledge with Semantic Web technology, creating support for significant portions of the semantic stack. Lucene provides document indexing and retrieval, enabling us to search through document collections from within applications. The ability to create a personalized index using AIDA enables customized document management and search on personal document collections using the same interface as that used for larger public collections such as MedLine. The machine learning techniques that have been applied within AIDA provide application builders with entity recognition for bio-entities such as proteins, as well as relation extraction ("proteinA *interacts with* proteinB") within text. The main bottleneck in training the statistical models that comprise our machine learning components is annotated training sets. This bottleneck points to the conundrum of how to produce annotated training sets without tedious manual annotation (impractical). Ideally, annotation would be performed automatically by employing the very same statistical model that one would like to train. This points to a natural progression of functionality: once you can search (for knowledge resources such as vocabulary terms), you can proceed to annotate, and once you have annotation, you can proceed to (machine) learn. This is where the knowledge resource tooling of AIDA can come in handy: annotators can use AIDA to search a number of repositories for the knowledge resources that they then use to annotate a given word or phrase. The resulting annotations could then be used to train a statistical model, as required for this approach to machine learning.

Although each technology is useful on its own, it is the combination of the aforementioned technologies that make AIDA truly useful. For example, in a well-established approach to text mining, the services that have been created from machine learning algorithms can be used to extract information from documents that have been retrieved from a Lucene index with information retrieval techniques such as query expansion (i.e., adding related terms to the query that increase recall). In a new approach to resource management, a selection of indexes (both personalized and public) can be searched with terms from vocabularies and ontologies that are made available from the RDF repository interface. This makes it possible to create a concept-based query of a personal resource collection, in which it is possible

to switch to other terminologies. Alternative terminologies and ontologies not only provide possible terms for queries but also for annotation.

It is the programmatic access to term-concept mappings (in our case, we make use of labels that are part of the RDF – more elaborate schemes are possible), as well as the ability to create them by annotation that lies at the core of personalized search. The ability to access vocabulary and ontological terms for both search and annotation enables the interface to be customized to the user. When mappings from concepts to multiple query languages have been made, a customizable user interface will become available for the browsing, management, search, and annotation of a wide range of data types and documents. Furthermore, the current machine learning web services could eventually be supplemented with other entity recognizers and types of relation extraction and even other forms of pattern recognition, including those from the field of image processing. This would make it possible to apply the same architecture in order to link concepts to other types of data, such as images, through the features pertinent to that domain that allow us to classify to certain concepts (i.e., tumor).

A few formidable challenges for the Semantic Web remain: end users would like to pose questions without having to know special query languages or know technical specifics such as where the information can be found. Technical solutions to this challenge exist, but will require much more work in the areas of query federation and repository annotation. Once the foundations have been laid, questions that are posed in the terms of (RDF) vocabularies can be translated to the form of a SPARQL query, where the query is automatically decomposed into subqueries that are dispatched to the data sources that contain the relevant data, the answers are assembled and presented to the user in a single application. Such a one-stop-shop will require sophisticated query federation, repository annotation, and software engineering. Some initial work in this area has been reported by the HCLS Interest Group [30].

Another important challenge for Semantic Web is to make it easier for people to use knowledge bases. When a user has access to a knowledge base, even if she is very knowledgeable about SPARQL, she will have to issue a SPARQL query in order to find out what can be matched as a subject, predicate, or object in subsequent queries. The process of discovering which items are available and which properties or predicates refer to them is a tedious and error-prone excercise. An interface that allows users to start from keywords and find the closest related terms in the knowledge base will lower this barrier considerably.

At the core of semantic e-Science is semantic disclosure. In order to enable computational experiments for biology research to be conducted in terms of concepts, with transparent access to workflow and grid resources such as data, knowledge models, and (web)services, we must make the disclosure of semantics and data provenance an essential part of experimental data production. It is our hope that this will convert databases and repositories from “data graveyards” into re-useable data pools, adding value to the data that then serves as evidence linked to the assertions in knowledge models.

How far are we in reaching the goals of semantic e-Science? A new era of data sharing has quietly begun. Enough organizations have opened up their data and

API's, with the XML data format becoming standard practice, that data exchange has become a trivial exercise. Service-Oriented Architectures (SOA) have also made it possible to share expertise in the form of programs as well as their components [31], such as the Taverna (and other) workflows that are shared on myExperiment [32]. BioCatalogue [33] will enable the community that employs these services to look up and report the quality and behavior of the services, as well as share information about how to use them. However, in general, knowledge sharing is still quite exceptional. Although it is simple enough to look up the syntactic type of a given piece of data (i.e., Integer, Float), there is generally no provision for semantic types (i.e., Chromosome Number, Score). This Do-It-Yourself (DIY) approach to semantics means that the handling of semantics is left up to the application developer who is building on the data and services. Until information systems provide the facilities to supply the semantics of a given piece of data, developers will continue to code assumptions about semantics into their programs and applications. Semantic support is especially important for workflow systems, where computational experiments can produce new knowledge but must store the new knowledge with labels that indicate its origins. Without integrated support for semantics in workflow systems, individual users must be motivated and knowledgeable enough to come up with their own ad hoc systems for disclosing metadata and knowledge from workflow components. In this respect we look forward to integrating the semantic approaches described in this chapter with the RDF-based provenance that is being developed for new versions of Taverna [34]. We look further toward an e-Science environment where semantics is just as much a fixed component as data and services, and knowledge about data and services, can be used to distribute jobs across grid nodes and partition data accordingly.

There are distinct benefits to collaborative research provided by the technologies used to build the e-Science applications described in this chapter. The combination of platform independent technologies such as SOAP, WSDL, Java, and Ajax in SOA-based applications has profoundly enhanced our ability to collaborate with colleagues. Typical problems that require communications overhead such as obtaining the latest version of code and compiling have been solved by simple web-based access to web services. We were able to take advantage of remote collaboration via web services in several ways. In the collaboration with Food Scientists, food vocabularies could be served to all interested parties and the latest version of the web interface that made use of those vocabularies via web services could be remotely evaluated by partners and testers without requiring any extra steps. A synonym server was made available to us as a web service by a partner in Erasmus University in Rotterdam and the service was effortlessly incorporated in our text mining workflow. The services available from AIDA have been updated without change to the API, allowing legacy applications to continue functioning.

How does our progress in semantic disclosure by the AIDA toolkit and others relate to high-performance computing, such as enabled by a grid? Web services have not yet been integrated with grid services in such a way that it is straightforward to combine them in an application (as is now possible with web services in many programs such as Taverna, Galaxy). At the time of writing, there is an artificial division

between grid and the Web largely due to data transport and security differences. For example, the default data transport for web services is SOAP, a protocol that wasn't designed for large data volumes or high throughput. Also, there is apparently no simple way yet to proxy grid credentials in order to provide web services access to grid services. The problem of data transport serves as an objection to web services for bioinformatics practitioners who are already processing large data in their programs. We expect that both the data transport and the security technical barriers to web and grid integration will be addressed in the short term. We are already applying one possible approach that has been implemented as a library that provides alternative data transport protocols through proxy. This approach has allowed us to distribute Lucene indexing over DAS[20] cluster nodes, which we expect will eventually speed up our nightly Medline indexing process and other large indexing jobs.

Despite the technical hurdles still to overcome, one of the most prominent challenges for e-Science is not of a technical nature but is related to the gaps in culture between the many fields involved in this multidisciplinary discipline, in particular the gap between application sciences and computer science. We think that it is important that application scientists try new developments in computer science at an early stage, providing feedback from real-life practice while application scientists can be the first to reap the benefits of a new approach. In our experience, however, a computer scientist's proof-of-concept is often not practically usable by scientists from the application domain. The theory might be proven academically by the computer scientist, but not implemented far enough to judge whether it is useful in practice, i.e., in application to a particular domain. In our view, one of the aims of e-Science is to overcome this gap. We believe that it is helpful to anticipate enough software engineers to create the necessary proof-of-concept implementations during the budgeting and planning stage of e-Science projects. We advocate their explicit addition to e-Science projects by making the analogy to wet laboratories, where scientists are typically supported by laboratory assistants who are trained at putting theory to practice. In addition, we have experienced that many computer scientists tend to apply the paradigm of "separation of concerns" to multidisciplinary collaborations, preferring to remain "domain agnostic" with the intent of providing only generic solutions. In our experience, this is often counter-productive. Without substantial understanding of each other's domains there is a risk that the mutual benefit is small or even negative. In fact, it is impossible to demonstrate the benefits of a new technology to a particular domain without carefully fitting it to an appropriate problem or use case. However, when these social factors of e-Science are taken into account, we are convinced that we will see some highly needed breakthroughs in, for instance, life science and health care.

Acknowledgments This work was carried out in the context of the Virtual Laboratory for e-Science project (http://www.vl-e.nl). This project is supported by a BSIK grant from the Dutch Ministry of Education, Culture, and Science (OC&W) and is part of the ICT innovation program of the Ministry of Economic Affairs (EZ). Special thanks go to Bob Herzberger, who made the VL-e

[20] http://www.cs.vu.nl/das3/

project a reality and to Pieter Adriaans for creating and leading AID. We also thank Edgar Meij, Sophia Katrenko, Willem van Hage, Kostas Krommydas, Machiel Jansen, Marten de Rijke, Guus Schreiber, and Frank van Harmelen. Our VL-e Food Informatics partners: Jeen Broekstra, Fred van de Brug, Chide Groenouwe, Lars Hulzebos, Nicole Koenderink, Dirk Out, Hans Peters, Hajo Rijgersberg, Jan Top. Other VL-e colleagues: Piter de Boer, Silvia Olabarriaga, Adam Belloum, Spiros Koulouzis, Kasper van den Berg, Kamel Boulebiar, Tristan Glatard. Martijn Schuemie, Barend Mons, Erik van Mulligen (Erasmus University and Knew Co.). Simone Louisse for careful reading of this document. Thanks to Alan Ruttenberg and Jonathan Rees of Science Commons for supplying the Huntington's corpus. We appreciate the support of many colleagues at NBIC, theW3C HCLS IG, myGrid, myExperiment, and OMII-UK.

References

1. Wang, X., Gorlitsky, R., Almeida, J.S.: From XML to RDF: How semantic web technologies will change the design of 'omic' standards. Nature Biotechnology 23 (2005) 1099–1103
2. Allemang, D., Hendler, J.: Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL Morgan Kaufmann (2008)
3. Stein, L.D.: Towards a cyberinfrastructure for the biological sciences: Progress, visions and challenges. Nature Reviews 9 (2008) 678–688
4. Galperin, M.Y.: The molecular biology database collection: 2008 update. Nucleic Acids Research 36 (2008) D2–D4
5. Ruttenberg, A., Clark, T., Bug, W., Samwald, M., Bodenreider, O., Chen, H., Doherty, D., Forsberg, K., Gao, Y., Kashyap, V., Kinoshita, J., Luciano, J., Marshall, M.S., Ogbuji, C., Rees, J., Stephens, S., Wong, G.T., Wu, E., Zaccagnini, D., Hongsermeier, T., Neumann, E., Herman, I., Cheung, K.H.: Advancing translational research with the semantic web. BMC Bioinformatics 8(3) (2007) S2
6. Marshall, M.S., Prud'hommeaux, E.: A Prototype Knowledge Base for the Life Sciences (W3C Interest Group Note). 2008 (2008)
7. Samwald, M., Cheung, K.: Experiences with the conversion of SenseLab databases to RDF/OWL (W3C Interest Group Note). Vol. 2008 (2008)
8. Ruttenberg, A., Rees, J., Samwald, M., Marshall, M.S.: Life sciences on the semantic web: The neurocommons and beyond. Briefings in Bioinformatics 10 (2009) 193–204
9. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. The Semantic Web – ISWC 2002: First International Semantic Web Conference, Vol. 2342/2002. Springer, Berlin, Heidelberg, Sardinia, Italy (2002) 54
10. LingPipe 4.0.0. http://alias-i.com/lingpipe (accessed October 1, 2008)
11. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, San Francisco (2005)
12. Katrenko, S., Adriaans, P.: Using Semi-Supervised Techniques to Detect Gene Mentions. Second BioCreative Challenge Workshop (2007)
13. Katrenko, S., Adriaans, P.: Learning Relations from Biomedical Corpora Using Dependency Trees. KDECB (Knowledge Discovery and Emergent Complexity in BioInformatics), Vol. 4366 (2006)
14. Koenderink, N.J.J.P., Top, J.L., van Vliet, L.J.: Expert-based ontology construction: A case-study in horticulture. In: Proceedings of the 5th TAKMA Workshop at the DEXA Conference (2005) 383–387
15. Rodgers, S., Busch, J., Peters, H., Christ-Hazelhof, E.: Building a tree of knowledge: Analysis of bitter molecules. Chemical Senses 30 (2005) 547–557
16. Rodgers, S., Glen, R.C., Bender, A.: Characterizing bitterness: Identification of key structural features and development of a classification model. Journal of Chemical Information and Modeling 46 (2006) 569–576

17. Smith, S.M.: Overview of fMRI analysis. The British Journal of Radiology 77(2) (2004) S167–S175
18. Olabarriaga, S.D., Boer, P.T.d., Maheshwari, K., Belloum, A., Snel, J.G., Nederveen, A.J., Bouwhuis, M.: Virtual lab for fMRI: bridging the usability gap. In: Proceedings of the 2nd IEEE International Conference on e-Science and Grid Computing, Amsterdam, Netherlands IEEE Computer Society, Los Alamitos, CA (2006)
19. Kent, W.J., Sugnet, C.W., Furey, T.S., Roskin, K.M., Pringle, T.H., Zahler, A.M., Haussler, D.: The human genome browser at UCSC. Genome Res 12 (2002) 996–1006
20. Thomas, D.J., Rosenbloom, K.R., Clawson, H., Hinrichs, A.S., Trumbower, H., Raney, B.J., Karolchik, D., Barber, G.P., Harte, R.A., Hillman-Jackson, J., Kuhn, R.M., Rhead, B.L., Smith, K.E., Thakkapallayil, A., Zweig, A.S., Haussler, D., Kent, W.J.: The ENCODE project at UC Santa Cruz. Nucleic Acids Research 35 (2007) D663–667
21. Belleau, F., Nolin, M.A., Tourigny, N., Rigault, P., Morissette, J.: Bio2RDF: Towards a mashup to build bioinformatics knowledge systems. Journal of Biomedical Informatics 41(5) (2008) 706–716
22. Cheung, K.H., Yip, K.Y., Smith, A., Deknikker, R., Masiar, A., Gerstein, M.: YeastHub: a semantic web use case for integrating data in the life sciences domain. Bioinformatics 21(1) (2005) i85–i96
23. Dhanapalan, L., Chen, J.Y.: A case study of integrating protein interaction data using semantic web technology. International Journal of Bioinformatics Research and Application 3 (2007) 286–302
24. Lam, H.Y., Marenco, L., Shepherd, G.M., Miller, P.L., Cheung, K.H.: Using web ontology language to integrate heterogeneous databases in the neurosciences. AMIA Annual Symposium Proceedings, Washington, DC (2006) 464–468
25. Marshall, M., Post, L., Roos, M., Breit, T.: Using semantic web tools to integrate experimental measurement data on our own terms. On the move to meaningful internet systems 2006: OTM 2006 Workshops (2006) 679–688
26. Post, L.J., Roos, M., Marshall, M.S., van Driel, R., Breit, T.M.: A semantic web approach applied to integrative bioinformatics experimentation: A biological use case with genomics data. Bioinformatics 23 (2007) 3080–3087
27. Boncz, P.A., Kersten, M.L., Manegold, S.: Breaking the memory wall in MonetDB. Commun. ACM 51 (2008) 77–85
28. Verschure, P.J.: Chromosome organization and gene control: It is difficult to see the picture when you are inside the frame. Journal of Cellular Biochemistry 99 (2006) 23–34
29. Hull, D., Wolstencroft, K., Stevens, R., Goble, C., Pocock, M.R., Li, P., Oinn, T.: Taverna: a tool for building and running workflows of services. Nucleic Acids Research 34 (2006) W729–W732
30. Cheung, K.-H., Frost, H.R., Marshall, M.S., Prud'hommeaux, E., Samwald, M., Zhao, J., Paschke, A.: A journey to semantic web query federation in life sciences. BMC Bioinformatics 10 (2009) S10
31. Miyazaki, S., Sugawara, H., Ikeo, K., Gojobori, T., Tateno, Y.: DDBJ in the stream of various biological data. Nucleic Acids Research 32 (2004) D31–34
32. De Roure, D., Goble, C., Stevens, R.: The design and realisation of the myexperiment virtual research environment for social sharing of workflows. Future Generation Computer Systems (2008) 2009 May, 25(5)
33. Goble, C., De Roure, D.: Curating scientific web services and workflows. Educause Review 43 (2008)
34. Missier, P., Belhajjame, K., Zhao, J., Goble, C.: Data lineage model for Taverna workflows with lightweight annotation requirements. IPAW'08, Salt Lake City, Utah (2008)

Chapter 3
A Smart e-Science Cyberinfrastructure for Cross-Disciplinary Scientific Collaborations

Hock Beng Lim, Mudasser Iqbal, Yuxia Yao, and Wenqiang Wang

Abstract Large-scale cross-disciplinary scientific collaborations are increasingly common and require an overarching e-Science cyberinfrastructure. However, the ad hoc and incoherent integration of computational and storage resources, sensor networks, and scientific data sharing and knowledge inference models cannot effectively support cross-domain and collaborative scientific research. In this work, we design and develop a smart e-Science cyberinfrastructure which forms the key resource-sharing backbone that enables each participating scientific community to expose their sensor, computational, data, and intellectual resources in a service-oriented manner, accompanied by the domain-specific knowledge.

3.1 Introduction

Science is broadly categorized into various distinct fields. However, there is a recent trend toward large-scale cross-disciplinary scientific research projects involving multiple organizations. A good example is in the field of environmental science, which involves studying the interactions of the physical, chemical, and biological components of the environment, with particular emphasis on the impact of human activities on biodiversity and sustainability. The participating researchers in such cross-disciplinary scientific research projects require efficient access to geographically distributed sensors and instruments, computational servers, and storage to carry out time-critical data collection, processing, analysis, and management tasks.

e-Science is a paradigm shift in scientific research that enables scientists to generate, process, and access data from distributed sources while making use of

H.B. Lim (✉)
Intelligent Systems Center, Nanyang Technological University, Nanyang Avenue, Singapore
e-mail: limhb@ntu.edu.sg

H. Chen et al. (eds.), *Semantic e-Science*, Annals of Information Systems 11,
DOI 10.1007/978-1-4419-5908-9_3,

shared resources of computing, storage, and virtual laboratories. Existing efforts in e-Science cyberinfrastructure aim to achieve the following objectives:

- Harness high-end computational resources available at distributed facilities.
- Access archived data and other intellectual resources from heterogeneous scientific databases.
- Make the resources belonging to an organization accessible to other users via secured and authenticated interfaces.

Although achieving these goals can help the scientific community to collaborate, the type of collaborations enabled is at a very coarse level to deal with the heterogeneous standards that are used by different scientific communities to manage their computing, data, storage, and intellectual resources. Such limitations hinder the pace of scientific discovery as it requires human intervention to resolve issues that arise due to the incoherent standards. For e-Science cyberinfrastructure to become a true backbone for seamless collaboration between different scientific communities, it should provide support to achieve the following additional goals:

- Publish and use the processes created in different scientific efforts in order to allow the creation of more sophisticated and complex processes.
- Correctly interpreting data from domains that are, by provenance, different from the one that is initiating the scientific process.
- Seamless integration with sensor and actuator networks to manage real-time sensor data.

There are various attempts that address one or more of the above goals using existing technologies available to scientific communities. For instance, grid computing [1] makes high-end computational as well as data resources available to scientific processes. Service-oriented architectures (SOA) provide a standards-based approach for making scientific processes from one domain available to other domains as a service. Semantic Web and ontology design enable the binding of data semantics with the data for ease of sharing and correct interpretation. In the following section, we discuss what these technologies aim to achieve and highlight various contemporary research attempts that aim to achieve one or more of the above-mentioned objectives. However, none of the existing efforts provide a single backbone that can support the entire spectrum of requirements that are essential to undertake effective cross-disciplinary scientific collaborations.

Increasingly, scientific communities are making information tools accessible as services that clients can access over the network, without knowledge of their internal workings. Service-oriented and grid-enabled e-Science provides support for dynamic selection of partners as well as abstractions through which the state of a scientific transaction can be captured and flexibly manipulated. In this way, dynamic selection is exploited to yield application-level fault tolerance. Services provide higher level abstractions for organizing applications in large-scale and open

environments. They enable us to improve the productivity and quality of the development of software applications. Furthermore, if these abstractions are standardized, they enable the interoperability of software produced by different scientific communities.

In this work, we propose a smart e-Science cyberinfrastructure that integrates grid computing, service-oriented architecture, and semantic Web in an innovative manner. The proposed framework builds a high-level ontology for this cyberinfrastructure that models sensor, computational, data, and intellectual resources as services that can be published and accessed. Figure 3.1 shows the proposed framework in the form of a smart e-Science fabric that binds together services, semantics, and grid with the following characteristics:

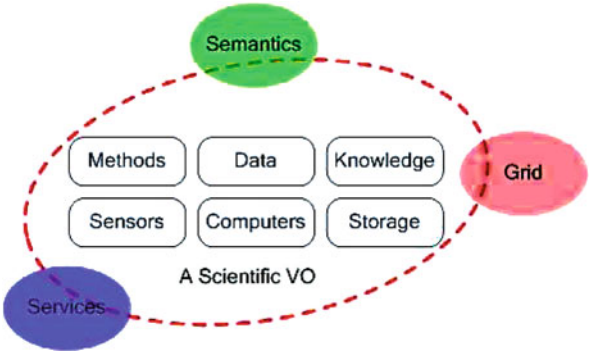


Fig. 3.1 Service-oriented e-Science fabric

- Use of grid middleware to handle state, transaction, notification, execution, monitoring, and scalability of the smart e-Science cyberinfrastructure.
- Semantic and rule-based event-driven SOA to demonstrate how semantics can be employed in SOA to share common vocabulary, knowledge, and services.
- Use of SOA for the composition of services to support development of scientific workflows.

In the rest of this chapter, Section 3.2 provides the background of this work. We describe the challenges in smart e-Science and the enabling technologies for smart e-Science. We also discuss the contemporary efforts in e-Science cyberinfrastructure and our case study to integrate various scientific infrastructures to form a smart e-Science cyberinfrastructure. In Section 3.3, we discuss our proposed smart e-Science framework in detail. In Section 3.4, extensive implementation details of the proposed framework are presented. Finally, we conclude this chapter and outline the direction of our future work in Section 3.5.

3.2 Background

3.2.1 Challenges for Smart e-Science

Although smart e-Science promises new vistas of scientific discovery by supporting collaborative scientific processes that span many disciplines, it poses many challenges:

- In collaborative scientific processes, it is not appropriate to expect the consistency of data from diverse sources that are parts of different scientific domains. However, it would be reasonable to think of the high-level contractual relationships through which the interactions among the components are specified.
- Such processes are usually so complicated that they cannot depend on the details of the implementations of the interacting components. Instead of modeling actions and interactions at a detailed level, it would be better to capture the essential high-level qualities that are (or should be) visible for the purpose of carrying out the process. Such a coarse granularity reduces dependencies among the participants and reduces communications to a few messages of greater significance.
- Individual components in the process must exist long enough to be able to detect any relevant exceptions, to take corrective action, and to respond to the corrective actions taken by others. Components must exist long enough to be discovered, to be relied upon, and to engender trust in their behavior.
- The concerns between the construction and the hosting of scientific resources are separated, so that scientists can focus on constructing content rather than on the minutiae of operating scalable and robust facilities.
- An open data-rich information system requires a semantic information hierarchy and a set of semantic services to process and reason about data, moving beyond just protocol agreement and data format conversion.
- More and more scientific organizations publish and access their scientific data using Web technology. Any scientific process requires a set of services to fulfill its goal. If this goal is not satisfied by a single resource then there is a need to combine the functionality of a set of resources as composite resources to fulfill the requirements of any complex scientific process. Distributed heterogeneous resources are to be integrated to complete the execution of a scientific process, and interoperability among resources is to be achieved when a process spans across the boundaries of multiple scientific domains, where vocabulary is different.
- The immense amount of scientific data from heterogeneous domains demands computing solutions that can use high-quality and domain-specific metadata in order to automatically interpret, integrate, and process data. Such solutions bring real value to scientists by answering domain-specific queries effectively to support knowledge discovery over large volumes of scientific data.
- Human mediation is inadequate to process, analyze, integrate, store, and query the petabytes of data and associated metadata generated by the industrial-scale

processes in e-Science. For software agents to be able to use this data, they must be able to interpret it correctly in the right context.
- Much of the scientific data are analogous in nature and are characterized by similar calibration mechanisms but described using different terms. String matching search techniques may not retrieve all the relevant data because different words/terms have been used to describe analogous processes in different scientific domains.
- In addition to archived scientific data, real-time data from actively operating infrastructures such as networks of environmental, seismic, and astrological sensors are very important. Provision of access to such networks is a challenge due to non-standard and domain-specific implementations and protocols.

3.2.2 Enabling Technologies for Smart e-Science

3.2.2.1 Grid Computing

Grid computing [1, 2] provides support for large-scale scientific and enterprise applications by allowing runtime selection, integration, and coordination of distributed resources and also accommodates dynamic requirements. It gives scalability and flexibility by following open standards, protocols, and technologies like Web services. The modern grids are based on open grid services architecture (OGSA) and Web services resource framework (WSRF), which extend the existing Web services and make them stateful, transient, and give notification support. They have become one of the standard solutions for scientific applications [3–5]. They enable the sharing and aggregation of millions of resources (e.g., SETI@Home [6]) geographically distributed across organizations and administrative domains. They consist of heterogeneous resources (PCs, work-stations, clusters, and supercomputers), fabric management systems and policies (single system image OS, queuing systems, etc.), and applications (scientific, engineering, and commercial) with varied requirements (CPU, I/O, memory, and/or network intensive).

3.2.2.2 Service-Oriented Architecture and Web Services

The importance of service-oriented architecture (SOA) for e-Science is widely recognized. SOA is the software architecture to enable loosely coupled integration and interoperability of distributed heterogeneous system by using services as component elements. Services are computational entities that can be described, published, discovered, orchestrated, and invoked by other software entities. An SOA usually includes directory services that service providers register with. Even if no adequate services are found in a service registry, it is still possible to build *composite* services by combining several pre-existing services.

Web services (WS) can also be used to publish, discover, and invoke the software components as services. They are loosely coupled, interoperable and enable the

integration of distributed heterogeneous Web-hosted services. Web services are based on open standards and protocols like XML, WSDL, UDDI, and SOAP [7, 8].

3.2.2.3 Semantic Web and Ontology

Semantic Web [9] is an attractive solution for organizing meaningful information. In particular, it is expected to revolutionize the real-time publishing and sharing of scientific information. This semantic provenance imposes a formally defined domain-specific conceptual view on scientific data, mitigates or eliminates terminological heterogeneity, and enables the use of reasoning tools for knowledge discovery. Ontology [10] is the key technology behind semantic Web for making information more meaningful, by adding more knowledge. The term ontology can be defined as an explicit formal specification of the knowledge by a set of concepts within a domain and the relationships between those concepts. Ontology binds together various concepts and makes sure that they are consistent and not overlapping. It is an explicit formal specification of a shared conceptualization [11].

An ontology consists of three components: (1) classes (or concepts) that may have subclasses to represent more specific concepts than in superclasses, (2) properties or relationships that describe various features and properties of the concepts, also termed as slots that are superimposed on the defined classes and/or properties to define allowed values (domain and range), and (3) individuals which are simply the instances of the classes and properties. The ontology, together with a set of instances of classes and slots, constitutes the knowledgebase. Many advantages of ontologies have been identified, including sharing common understanding of the structure of information, enabling reuse of domain knowledge, making domain assumptions explicit, separating domain knowledge from operational knowledge. The ontology development follows an evolving-type life cycle rather than a waterfall or an iterative one. This implies that one can go from one stage to another stage in the development process if the ontology does not satisfy or meet all the desired requirements.

Ontology is based on metadata whose critical role in managing large volumes of data has long been understood in various fields such as library management, geography, multimedia, biological sciences. The database community has extensively explored the use of metadata to exchange, share, and integrate data from heterogeneous information sources. Since traditional metadata descriptions (such as electronic data-interchange formats) require manual interpretation, researchers have proposed using semantic metadata to automate the integration of large-scale distributed data. Semantic metadata is "metadata that describes contextually relevant or domain-specific information about content (optionally) based on an ontology." It not only mitigates terminological heterogeneity but also enables software applications to "understand" and reason over it. Ontologies can thus provide a natural means for representing information for database management systems because databases may have inconsistencies in naming conventions, synonyms, etc. Thus, having a

common ontology for each application or scientific domain helps to unify information presentation and permits software and information reuse.

3.2.3 Contemporary Efforts in e-Science Cyberinfrastructure

There have been various attempts to address the collaborative science challenges. We will provide an overview of some of these existing efforts. TeraGrid [12] is one of the foremost cyberinfrastructure in the scientific community. It comprises of immense amounts of geographically distributed computational and data resources that are accessible in a service-oriented manner. For job management (submission, workflow, etc.), it provides a set of software packages that can be used by the users to remotely administer their jobs on the resources that have been allocated to them. Although its user base is large, the TeraGrid does not provide a facility for seamless interoperability between the various scientific communities utilizing it. Essentially, it only provides grid resources to the end users and a service-oriented means of managing them while the operational and architectural challenges pertaining to the use of TeraGrid for cross-disciplinary science are left to the end users.

GEO Grid [13] aims at providing e-Science infrastructure specifically for earth science communities. It provides support for satellite imagery, geological, and ground sensor data. Architecturally, GEO Grid provides different levels of interactions among its users depending upon the role that the user wishes to play. At the source sits a data publisher that can be any earth science community who can provide data in a service-oriented manner. A virtual organization (VO) manager forms a virtual portal where he can request services from different data publishers and provide to the end users. A VO manager may compose and provide complex services by using the basic services provided by data publishers. The key limitation of GEO Grid is that it mainly aims to serve the earth science community. In addition, although GEO Grid allows data publishers to specify metadata, it requires them to implement data access services which have a high risk of running into non-standard data semantics and access methods.

The London e-Science Center (LESC) [14] is an effort similar to GEO Grid that also provides VO-based access to integrated services from multiple service providers. However, LESC's scope is far broader than that of the GEO Grid in terms of supporting different scientific communities. Both GEO Grid and LESC are limited in terms of not providing any tools or guidelines for standardizing data and resource ontologies to ensure precise interpretation of data. While the VOs may serve as a platform to access integrated data and services, the absence of standardized ontology definition and service definition framework makes the job of the VO manager (who is going to compose new services) very difficult.

The Cambridge e-Science Center (CeSC) [15] aims to develop grid-based tools for massive data handling, high-performance computing, and visualization applications. The infrastructure runs Globus on Condor for managing workloads of compute-intensive jobs and MyProxy for credentials management. In terms of grid

service provisioning, CeSC is similar to TeraGrid, with both of them providing no framework for collaborative e-Science on the grid.

The OntoGrid [16] addresses this greatest challenge faced by grid computing regarding the ability to explicitly share and deploy knowledge to be used for the development of innovative grid infrastructure and for grid applications. To address this challenge the OntoGrid project aims to produce the technological infrastructure for the rapid prototyping and development of knowledge-intensive distributed open services for the semantic grid. However, the current test beds and implementations of OntoGrid do not demonstrate cross-disciplinary data exchange and interoperability.

The MyGrid [17] effort aims to support the creation of e-laboratories to allow scientists share valuable data and laboratory resources. However, its current implementations are limited to same-domain workflow service creations and cataloguing [18].

The above-mentioned e-Science grid frameworks provide reasonably sufficient and scalable grid resource provisioning infrastructures to the scientific communities to carryout compute/data-intensive tasks. However, none of them provides a scientific resource-sharing framework at the infrastructure level on top of which the participating scientific communities can securely expose, discover, and exchange their intellectual (data and methods) as well as instrument-based (sensor networks, devices, laboratories, etc.) resources. In the absence of such a framework, the existing e-Science grids are just passive compute–storage resource providers. The smart e-Science framework proposed in this chapter fills this gap by providing a semantic resource-sharing framework that can be readily adopted by the existing e-Science grids.

3.2.4 Case Study: Integration of Scientific Infrastructures

To design and develop a smart e-Science cyberinfrastructure for cross-disciplinary scientific collaborations, we embark on a case study to integrate several scientific infrastructures that we are involved with. In this section, we describe these scientific infrastructures that drive the design and implementation of the proposed smart e-Science framework. These infrastructures involve large-scale sensing and monitoring deployments that produce high data rate heterogeneous scientific data.

The Cyberinfrastructure for the Center for Environmental Sensing and Modeling (CI@CENSAM) [19] is an effort under the Singapore–MIT Alliance for Research and Technology (SMART) to integrate the data and services provided by various micro-to-meso scale sensor deployments for environmental monitoring in Singapore. These deployments include those for Continuous Monitoring of Water Distribution Systems (CMWDS), Ocean Modeling and Data Assimilation (OMDA), and Marine and Underwater Sensing (MUS). CI@CENSAM aims to develop a distributed data archive for multiple CENSAM research projects including both sensor-

and model-generated data. The archive will associate data sets with appropriate geospatial, sensor, accuracy, and access control metadata to optimize the data′s utility, security, and longevity for collaborative scientific efforts.

The Continuous Monitoring of Water Distribution Systems (CMWDS) project will develop the technologies to enable real-time monitoring of water distribution systems in Singapore. Its objectives include the demonstration of the application and control of a wireless sensor network-based cyber-physical infrastructure for high data rate, real-time monitoring of hydraulic parameters within a large-scale urban water distribution system. Real-time pressure and flow measurements will be assimilated into hydraulic models to optimize the pump operations for the water distribution network. CMWDS also develop the technologies to enable remote detection and prediction of pipe burst and leak events. We have developed statistical and wavelet-based algorithms to analyze high-frequency pressure measurements of hydraulic transient events to detect pipe bursts, as well as algorithms for localizing the bursts based on arrival times of the pressure fronts associated with the burst events. Finally, CMWDS also address the monitoring of water quality parameters. This task will involve a detailed evaluation of the long-term performance and robustness of water quality sensors (for measures such as pH, chlorine residual, turbidity, conductivity, and dissolved oxygen), the use/development of multi-parameter sensor technologies, and the application of cross-correlation techniques to interpret water quality signatures through in-network processing.

The Ocean Modeling and Data Assimilation (OMDA) project aims to achieve operational real-time assimilation and forecasting capabilities for the Singapore region and surrounding seas. For this purpose, the Finite Volume Coastal Ocean Model (FVCOM) is used and is adapted for different configurations such as the coastal water around Singapore, the Singapore Straits and island, and the entire South China Sea (90E–140E; 20S–30 N). The simulation of the ocean circulation and property distributions (temperature, salinity) will be carried out under lateral tidal forcing and surface forcing of wind stress, heat, and moisture fluxes and will be validated with altimetric data as well as current velocity observations.

The Marine and Underwater Sensing (MUS) project's objectives include the development of sensors for environmental chemical monitoring that can be deployed on automated underwater vehicles (AUVs). A major focus of MUS is a sensor based on mass spectrometry for monitoring of natural waters, which measures low molecular weight hydrocarbons, metabolic gases for geochemical studies, and volatile organic compounds for pollution monitoring. The other major focus is a sensor based on laser-induced fluorescence, capable of measuring higher molecular weight hydrocarbons which are common components of oil leaks and spills, as well as biological entities such as chlorophyll, aquatic humic substances, and fluorescent tracers. An associated project under MUS will deploy inexpensive, low-power sensors for passively detecting dynamic and static pressure fields with sufficient resolution to detect near-field flow patterns and near- and far-body obstacles and vehicles, as well as mapping near-body objects. This will provide a unique capability for navigation in shallow water and/or cluttered environments, for use with multiple AUVs, and for flow control in conventional and biomimetic vehicles.

The National Weather Study Project (NWSP) [20] is a large-scale community-based environmental initiative in Singapore that aims to promote the awareness about weather patterns, climate change, global warming, and the environment. In this project, hundreds of mini weather stations are deployed in schools throughout Singapore. The data acquired by these weather stations include various weather parameters such as temperature, humidity, rain, wind speed and direction, barometric pressure, solar radiation.

The aim of the Solar-enabled Water Production and Recycling (SWPR) project is to develop and demonstrate self-sustaining water production and recycling technology based on solar energy. In this project, the solar radiation is one of the most important parameters for data collection. Comprehensive spatio-temporal solar radiation data analysis, data mining, and data modeling are carried out to identify suitable sites in Singapore for building pilot solar-enabled water production and recycling plants. The project also addresses the optimization of the operation of the solar-powered systems for the water production and recycling processes by considering the variation in solar radiation pattern with time.

The Sensor Grid for GPS Data Processing Project (GPS) [21] is an effort to develop a prototype sensor grid to support the acquisition, analysis, visualization, and sharing of GPS data in the earth science domain. As a test case, we make use of historical GPS data collected from the Sumatran cGPS Array, which consists of a network of GPS stations deployed in the high earthquake activity zone of Sumatra, Indonesia.

In the LiveE! Project [22] from Japan, a large-scale peer-to-peer network connecting environmental monitoring sensors is deployed across the Asia-Pacific region in Japan, Taiwan, China, Indonesia, etc. The data produced by this large network also comprises of various weather parameters.

Each of the deployments in CI@CENSAM, NWSP, SWPR, GPS, and LiveE! includes archived scientific data as well as real-time sensor resources with varied data sampling frequency and heterogeneous types of sensors provided by different vendors. In addition, the environmental, ocean, navigation, hydraulic, geospatial, solar, and seismic modeling applications running on top of these deployments require seamless sharing of scientific processes, data models, and raw data among them.

3.3 The Smart e-Science Framework

We propose a proxy-based approach for our semantic grid infrastructure. With this approach, the resources in a scientific facility (methods, data, storage, computational servers, etc.) are made available on the grid like conventional grid services. Figure 3.2 shows various components of the proposed grid infrastructure. The key entity in the infrastructure is a virtual organization (VO) that represents a resource-sharing facility in the grid. Each VO may provide one or more resources such as heterogeneous sensors and sensor networks, computational and storage resources, scientific methods and knowledge, and grid-enabled service providers. Typical

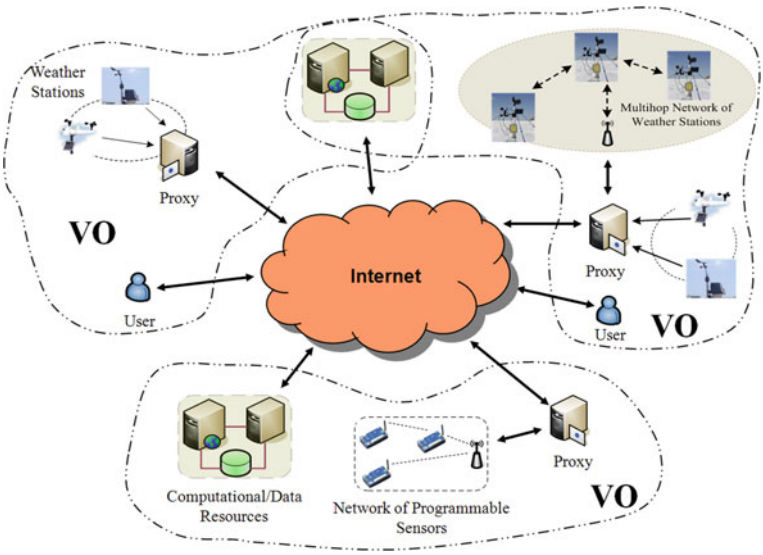


Fig. 3.2 Design components of the proposed infrastructure

examples of VOs are virtual laboratories, scientific data centers, supercomputing facilities, networks for metropolitan traffic monitoring, environmental monitoring, a smart home, an elderly care facility, a corporate office. From the above-mentioned case study, each of the infrastructures in CI@CENSAM, NWSP, SWPR, GPS, and LiveE! is thus a VO. The key idea is to enable each VO to access and share the resources of other VOs through distributed resource brokerage and user authentication. The proxy interface at each VO manages the VO-level ontology and exposes the resources in the VO as services, thus allowing for cross-domain interoperability.

3.3.1 The e-Science Ontology

In this section, we describe the e-Science ontology that allows semantic exposure of resources in VOs in the form of services. First, we construct an ontology based on the common vocabulary set in the participating VOs (CI@CENSAM, NWSP, SWPR, GPS, and LiveE!). Each VO will then create an instance of this ontology in its domain and share it with the rest of the VOs that belong to the grid. We term the VO that owns the resources as *server VO*, whereas the one making use of the resources as the *client VO*. The basis of this design strategy is the fact that it is not practical to standardize the way tangible (storage, servers, sensors) and intangible (intellectual knowledge, data, methods) resources are managed within heterogeneous scientific communities. However, the interfaces to access these resources from outside the server VO are still in infancy and thus must be standardized.

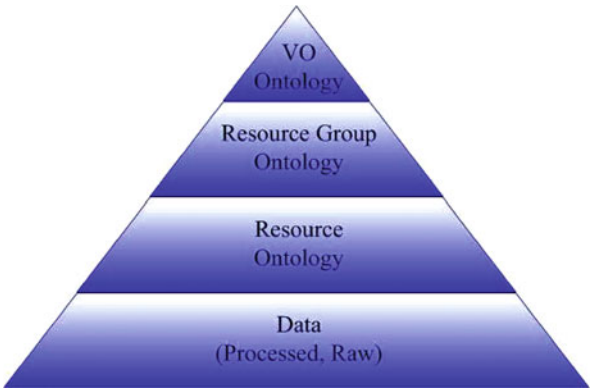


Fig. 3.3 The layered design strategy for e-Science ontology

Figure 3.3 shows the layered design of the proposed e-Science ontology. This design is motivated by the natural organization of resources and data in a VO, where a VO sits on top of the group of resources (ResourceGroup) which in turn defines each resource in the VO such as compute grid, sensor networks, data archives. Each resource then produces its own data which are exchanged between the participating VOs.

Figure 3.4 shows the ontology framework that emerges out of the triangle design. Figures 3.5–3.7 show the detailed ontologies of various components of the overall ontology in Fig. 3.4. The ontology considers that each VO runs some R&D programs that comprise of some projects. In addition, each VO is assumed to consist of *Resource Groups*, with the resource groups related to one or more projects. The resource group class is the highest level class in the ontology hierarchy with two immediate subclasses to categorize *physical* and *non-physical* resources in the VO. The physical resource superclass refers to all tangible resources such as computational servers, storage, sensors, and sensor networks. Each of these classes is then inherited by more specific physical resource types in order to explicitly elaborate the resource attributes.

On the other hand, the non-physical resource superclass refers to data (raw as well as processed) and scientific methods that can be made available for reuse to client VOs. The data and methods superclasses are then inherited by more specific data-related ontology hierarchy and methods-related process ontology hierarchy that define characteristics like the type, nature, format, interpretation, input, output, and the domain of data and scientific processes being expressed by the ontology. The Methods class hierarchy is of key importance in this e-Science ontology since it categorizes scientific methods into data management (acquisition, transfer, archival) and data processing (any scientific processes that result in data transformation). The process ontology that defines these classes of methods thus forms the basis for providing service-oriented access to intellectual resources in the VO and allows for creation of complex workflows. Another key advantage of binding methods in

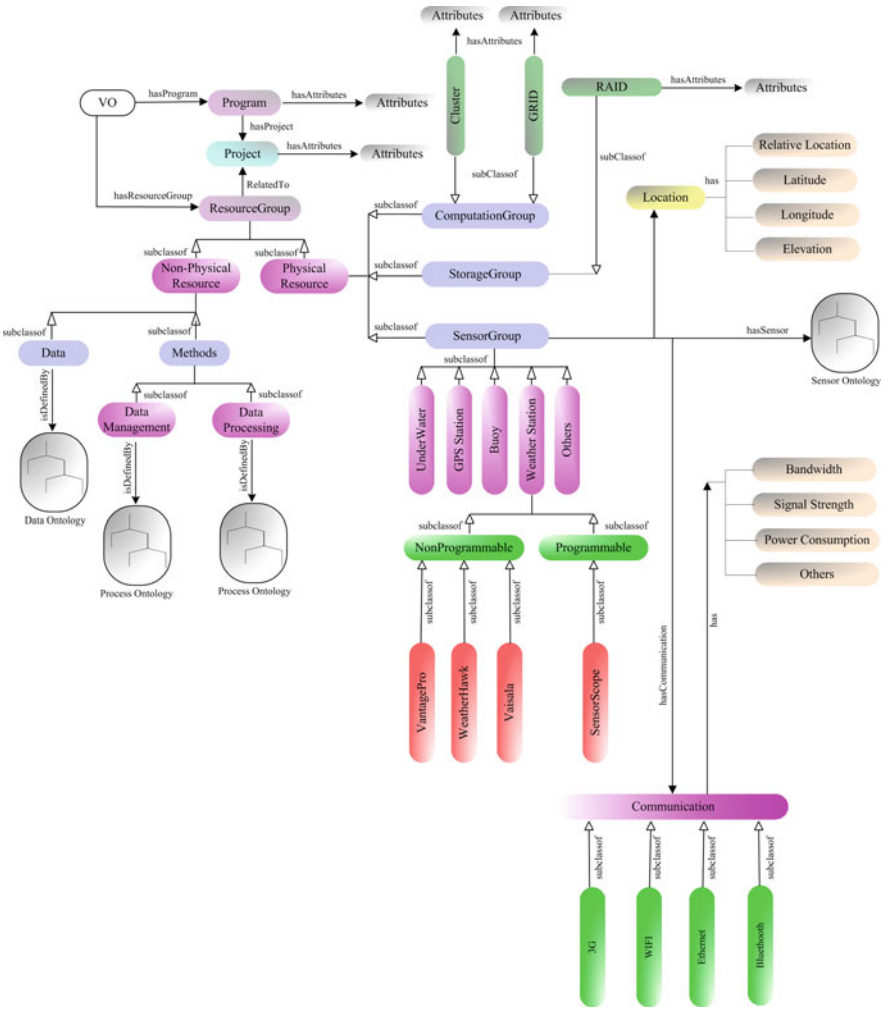
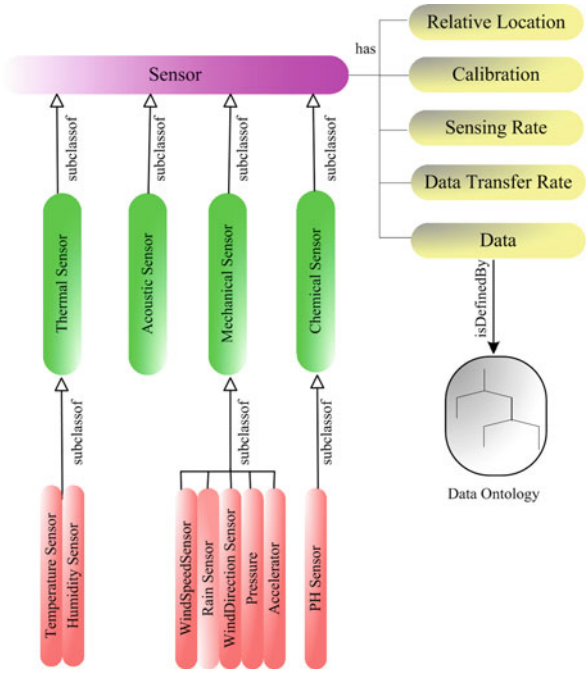


Fig. 3.4 The e-Science ontology

the same ontology hierarchy is that it allows operational standardization within the ontology, as each scientific method also becomes a subclass of a common ResourceGroup superclass. We will discuss this topic, commonly termed as service composition, in Section 3.3.2.

Since the purpose of defining an ontology is to make the resources shareable, it is the physical and non-physical superclasses hierarchy that should be laid down with maximum details. Due to space limitation, the detailed ontology for each of the subclasses in the ResourceGroup hierarchy is not provided. However, as an example, we have shown a part of the expanded SensorGroup ontology section that specifies the categorization of sensor groups into various domain-specific sensing units, such as

Fig. 3.5 The sensor ontology



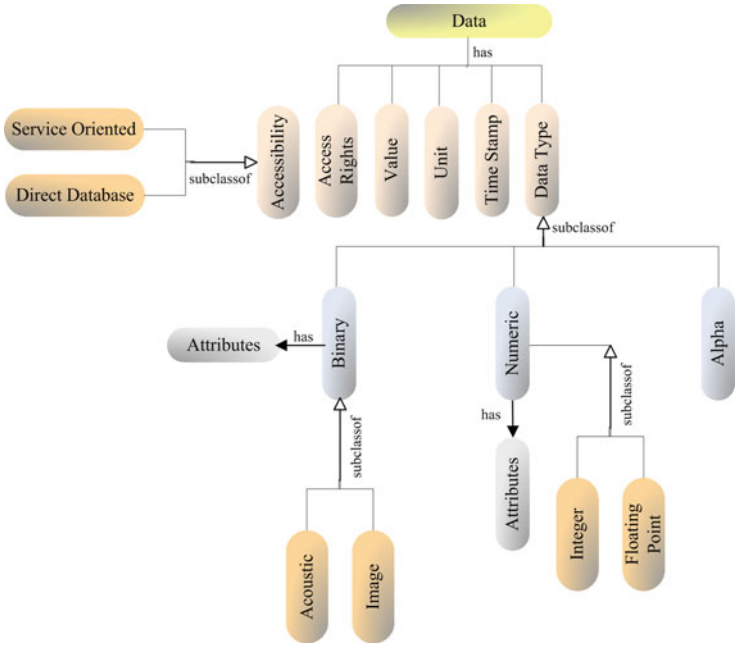


Fig. 3.6 The data ontology

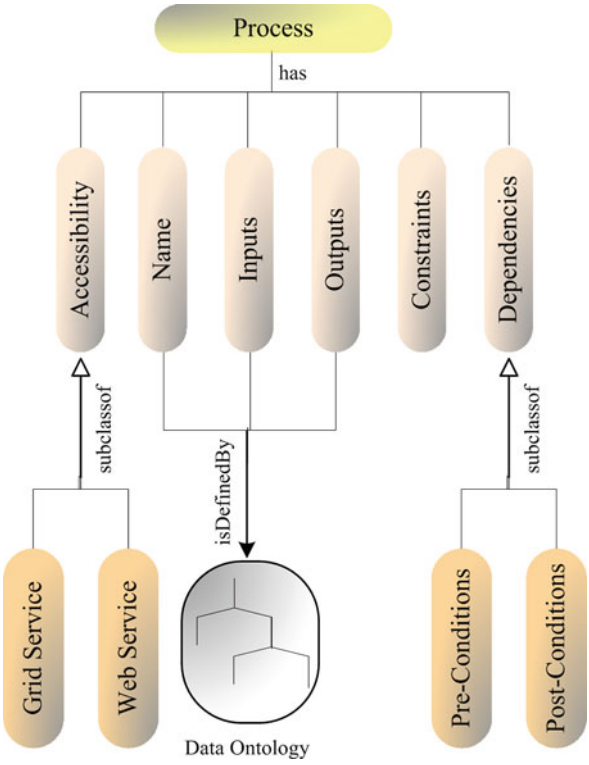


Fig. 3.7 The process ontology

weather stations, GPS stations, buoys and underwater sensors. A SensorGroup has also been shown to have an associated geographical location identified by latitude, longitude, elevation, and (if necessary) an *x*–*y* location relative to some reference.

Each SensorGroup may consist of multiple heterogeneous sensors with each sensor being identified by its attributes such as its calibration coefficients, rate of sensing, data transfer rate, and the data itself. The ontology also caters for situations where a SensorGroup may have multiple instances of the same sensor which are differentiated by their relative locations within the sensor group. For example, this feature can cater for underwater sensors that may carry multiple temperature sensors with each sensor responsible for monitoring water temperature at a different depth. The data attribute under the sensor ontology is further expanded to represent different data types such as alpha (which includes alphanumeric), numeric, and binary data. The numeric data can be any integer or real number quantity such as water pressure, pH value. The binary data represent sensors such as acoustics and imagery (such as satellite topographical images). Apart from the sensors, each SensorGroup may have other resources such as communication modules, battery power, and storage. Due to space limitation, only the communication module has been shown in the ontology in Figs. 3.4–3.7 with its possible subclasses such as 3G, WiFi, and Bluetooth, and its attributes such as available bandwidth and signal strength. The complete ontology is available at [23].

In defining the ontology, our main source for collecting commonly used terms in the sensor domain was the IEEE 1451.4 smart transducers template description language [24], Semantic Web for Earth and Environmental Terminology (SWEET) [25], Geography Markup Language (GML) [26], Sensor Web Enablement (SWE) [27], SensorML [28], Suggested Upper Merge Ontology (SUMO) [29], and OntoSensor [30], as well as domain-specific terminologies to handle our projects discussed in Section 3.2.4. The ontology is implemented in Protégé with the data ontology expressed in the Web Ontology Language (OWL) [31], whereas the methods ontology is expressed in the OWL-compliant OWL-S [32, 33] format. The detailed OWL descriptions of the respective terms are found in the above-mentioned ontologies. Specifically, the SWEET ontology may be consulted for scientific sensor measurements such as water pressure, acoustic profile, underwater temperature, relative humidity, whereas geographical and topographical ontology elements are available from GML.

3.3.2 Grid-Based Service Orientation

In this section, we discuss the service advertisement, discovery, and access mechanisms on the grid-enabled infrastructure. Essentially, resource consumers in a scientific community need a service-oriented model to allow them to specify resource requirements and constraints. They need brokers that provide strategies for choosing appropriate resources and dynamically adapt to changes in resource availability at runtime to meet their requirements. In addition, in service-oriented e-Science, there will be multiple, concurrent clients exercising different functionalities of the system for different scientific processes. Thus, the infrastructure must have a certain level of autonomy to decide on the best use of available resources to fulfill multiple users' concurrent uncoordinated requests.

Under these objectives, we argue that if the infrastructure provides means to access the ontologies, one may encapsulate sophisticated domain-specific knowledge and scientific methods into computational components or services. Each service simply transforms input events into output events, adding new semantic information as necessary. This semantic-based approach helps in search, discovery, selection, composition, and integration of services and also in the automation of their invocation, composition, and execution. Our proposed framework uses semantic services as the basic programming model. We call these components as semantic services since they provide access to various resource groups defined by the ontology of a VO. Figure 3.8 shows the architecture for grid-based semantic service management. The following sections provide details on each of the modules.

3.3.3 Service Interface

A scientist in a client VO needs to access resources in a server VO. In this case, the proposed infrastructure supports the most easily accessible means of access, i.e., Web services, over unsecure channels (http) or via secure grid interfaces (https).

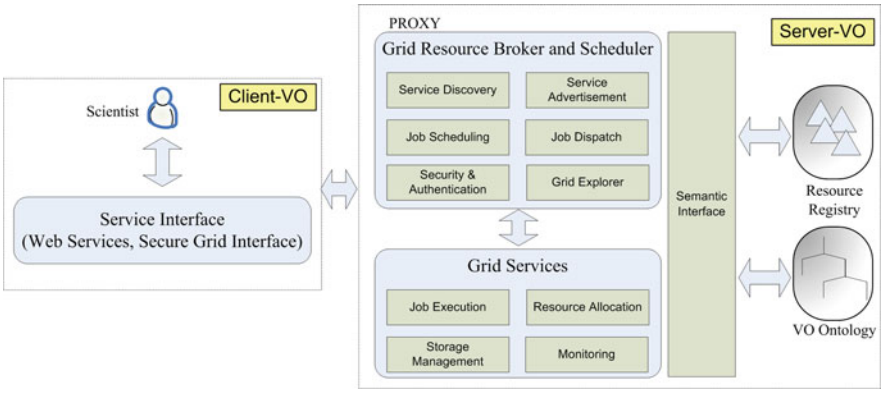


Fig. 3.8 The smart e-Science architecture

3.3.4 Semantic Interface

The semantic interface is a very important component of the entire infrastructure since it provides access to both the resources directory and their semantics (ontology). This component is implemented as a service that receives requests to locate particular concepts in the directory and retrieve their semantics from the ontology. The output from the semantic interface is always a list of semantic objects that describe their type (data, scientific method, sensor, storage, computer), the access mechanism (port type, address), and the interface (input, output).

3.3.5 The Proxy

The request from a client VO is received on the server VO side by the proxy node that hosts resource brokerage and scheduling mechanisms. The request is first authenticated via grid certificate credentials or directory-based third-party sign-on mechanisms. The job scheduling and job dispatch components employ the grid's middleware services (such as ManagedjobFactory) for scheduling execution of a service and dispatching it to the appropriate node on the grid. In order to find the appropriate node, the grid explorer component is used. This grid explorer component keeps grid node statistics via resource registry.

One of the key components residing at the proxy node is the service discovery component that exposes the resources (physical as well as non-physical) in the server VO to the outside world. Since the resources in the VO are modeled using an ontology, the service discovery component provides interfaces to each of the services listed in the resource directory, through which the services can be accessed. For the resources to be discovered, they are first advertised in the form of service ontologies by registering with the resource registry. While existing systems support an attribute-based discovery as well as a simple name lookup to locate a service, we exploit the OGSA's inter-grid interoperability and discovery mechanisms and

bundle them with semantic matching based on the server VO's ontology. This mechanism enables both the provider and the user to share a common understanding of what capabilities the service offers and how they can be put to use.

3.3.6 Knowledgebase Design

Designing an archive for a smart e-Science cyberinfrastructure is a challenge since the schema not only has to support high data rate Online Line Transaction Processing (OLTP) queries but also has to cater for modeling and decision support queries in the Online Analytical Processing (OLAP) domain. Thus, we designed the schema for our smart e-Science cyberinfrastructure with the following features:

1. It supports high rate incoming data streams that comprise of binary as well as discrete data types.
2. It supports real-time data retrieval (OLTP).
3. It supports large-scale data retrieval to interface with third-party modeling and simulation tools (OLAP).
4. It supports distributed data storage and processing in order to handle vast amount of sensor data.
5. It is extensible and flexible to handle arbitrary new sensor type.

3.3.7 Data Exchange

The triangle design shown in Fig. 3.3 forms the basis of seamless data exchange among the participating VOs. Starting from the lowest layer in the triangle, each instance of the triangle, when coupled with a data set, provides the semantics of the data itself, the resource that produced the data set, the resource group to which this resource belongs, and the VO that owns this resource group. However, we need a mechanism to let the participating VOs exchange these ontology triangles among each other whenever they intend to exchange data. In addition, this data exchange should conform to the data security and user access policies as defined by the VO owning the data.

The proposed smart e-Science framework allows such a data exchange using grid services that act like a middleman between the two VOs intending to exchange the data. From the perspective of the client VO (the one seeking data), Fig. 3.9 shows that the data emerging from heterogeneous scientific sources are interpreted by the e-Science grid middleware using the data source's ontology. The interpreted data are then transformed using the highest level data ontology into the format expected by the client VO, as described in the query specifications.

Figure 3.10 shows the step-by-step flow of events starting from the initiation of query from a VO (VO1) until it gets the interpreted and transformed data back from another VO (VO2). The flow begins with VO1 issuing a data acquisition request.

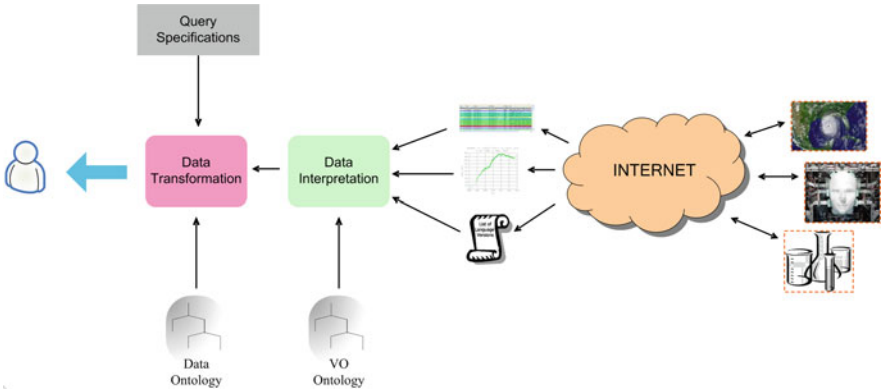


Fig. 3.9 Data flow from the perspective of client VO

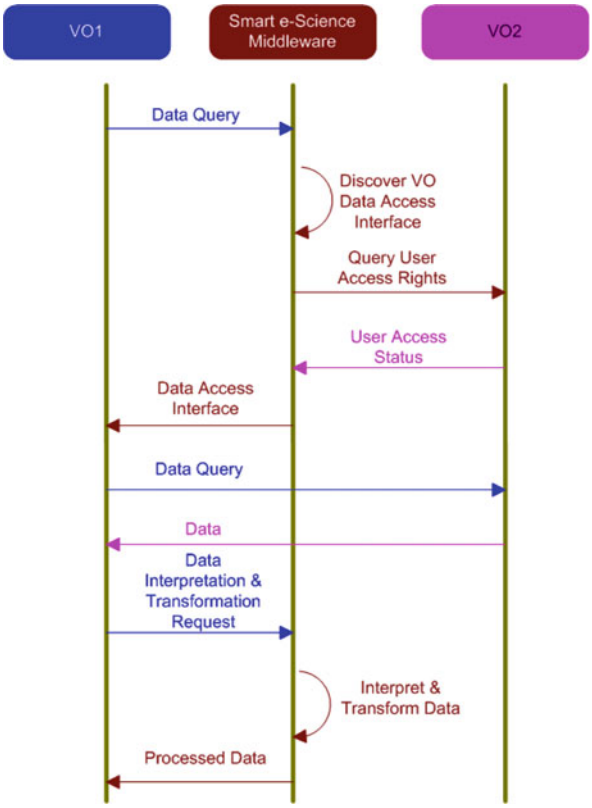


Fig. 3.10 The data exchange between two VOs

This request will include the high-level semantics regarding the type of data that is sought and, possibly, the target VO (if known). The e-Science middleware searches through the ontologies of all the participating VOs (if no target has been specified) in order to discover a data access interface. Once such an interface is found, the middleware contacts the target VO (VO2) to seek VO1's access rights. If VO2 clears VO1's access, the data access interface is returned back to VO1. This interface can be a link to a Web service, a database connection information, or a Web-based data archive. VO1 uses this interface to directly connect to VO2 seeking the data. VO2 responds back with the data set that satisfies VO1's query. However, this data set is incomprehensible to VO1 as it belongs to a different operational and scientific domain. Thus, VO1 sends a service query to the e-Science middleware to interpret the data and transform it into a format (such as XML, CSV) that is friendly to VO1. The e-Science middleware uses VO2's ontology to identify the ontology triangle for the data set and converts the data using the highest level data ontology. The reformatted data are then sent back to VO1 which then uses it for the purpose they were sought.

3.4 Implementation

Based on our sensor grid architecture design, we are building a large-scale smart e-Science cyberinfrastructure to integrate hundreds of heterogeneous sensor resources distributed across a wide geographical area under the CI@CENSAM, NWSP, SWPR, GPS, and LiveE! projects into a single semantics-based service-oriented access model. The details of our sensor grid design and implementation are described in [34].

Our sensor grid-based cyberinfrastructure has several important features. First, it connects heterogeneous sensor resources belonging to different scientific VOs via the Internet to automatically collect and aggregate sensor data in real time. Second, the varied types and formats of sensor data are linked to the ontology of the respective VO and the data are converted into a standard OWL format in order to bind data semantics with the data for ease of access and processing. Third, the sensor data are stored in distributed data archives whose schema is also derived from the common concepts found in the ontologies of three VOs. Fourth, the grid models computational as well as storage resources as semantic services for the compute-intensive processing of sensor data. Fifth, the sensor data can be conveniently accessed and shared via the Web through mash-ups, blogs, and Web services. We are developing techniques and tools to efficiently publish, archive, query, process, visualize, and search the vast amount of sensor data in a unified manner.

In the following sections, the details of the data and process ontologies, client and server VO service implementation, and the design of data schema are provided.

3.4.1 Ontology Design

We selected the Web Ontology Language (OWL) to implement various data and process ontologies since it is designed for use by applications that need to process the content of information instead of just presenting information to humans. OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with formal semantics. The OWL can be used to describe the classes and relations between them that are inherent in Web documents and applications. 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, i.e., facts not literally present in the ontology, but entailed by the semantics. The advantage of OWL over standard XML schema is that OWL is a knowledge representation and helps accurate interpretation through its rich semantics.

We used the Protégé OWL for implementing the guidelines for building ontologies for our case study domains. The steps to define this ontology are as follows:

1. All elements from the CI@CENSAM, NWSP, SWPR, GPS, and LiveE! projects were identified and conceptualized as classes. Table 3.1 shows a list of such concepts.
2. Various high-level concepts such as projects, ResourceGroup, data, and sensor were identified as root classes.
3. A scientific VO was considered as the origin of the ontology.
4. The relationships such as *has, subClassOf, isDefinedBy* between the root classes with the VO were identified.

Table 3.1 Concepts in an ontology for various scientific domains

Sensor	Sensor location	Average temperature
Weather station	Vantage Pro	Surface temperature
GPS receiver	Weather hawk	Rain alerts
LiveE!	Vaisala	GPS observation data
NWSP	SensorScope	Satellite navigation data
APEC	Earth plate movement	Satellite
Sumatra, Indonesia	Temperature	Manufacturer
Japan	Humidity	Model
Singapore	Wind speed	Array of GPS stations
Station location	Inside temperature	Data rate
Numbers	Outside temperature	GPS position estimation
Strings	School	GPS velocity estimation
Project	Data precision	Underwater depth
Elevation	Data format	Total chlorine
Building geometry	Leak	PH
Underwater vehicle	Pressure	Velocity
Chemical sensors	Sound	Orientation
	Surface wind	Ocean carbon
		Biological agents

5. Subclasses for each of the root class were identified with their attributes and laid in the ontology hierarchy.
6. Specific instances of various classes, such as WeatherStations, ThermalSensors, AcousticSensors, SuperComputer, RAID, were created.
7. The data and process ontologies were defined for each VO.

Table 3.2 shows the characterization of various concepts in the resulting ontology. A high-level view of the ontology for smart e-Science is shown in Figs 3.4–3.7.

Table 3.2 Characterization of concepts in an ontology

Domain	Property name	Range
Sensor_characterized_by	Sensor	Type, Max_Output, Min_Output, accuracy, location, datastream
SensorGroup_characterized_by	SensorGroup	Type, location, manufacturer, sensor
hasDataType	Datastream	Format
Project_characterized_by	Project	Identification, SensorGroup
Project_Canbe	Identification	Developer, name, starting_time, country
Location_Canbe	Location	Location_on_earth, Location_in_ SensorGroup
Locationonearth_Has	Location_on_earth	Latitude and longitude
LocationinGroup_Has	Location_ in_SensorGroup	Distance to the ground of the fix point
HasUnit	Format	Physical_Unit

3.4.2 Server-Side Service Implementation

As mentioned earlier, the server VOs will provide access to their resources via server-side services, whereas client VOs will utilize those services by deploying client-side services. For our smart e-Science cyberinfrastructure, we have implemented a number of services to expose both physical and non-physical resources. The server-side services have been implemented in Java and deployed in Apache Axis to produce server- and client-side SOAP wrappers. As an example, we will provide the implementation details of a sensor data retrieval service GetSensorData.

Each service comprises of methods that can be categorized based on the type of information they provide (a) metadata-related methods and (b) data-related methods. The metadata-related methods deal with obtaining meta information about the

service itself, such as the list of methods that the service provides and the signatures of those methods so that they can be accessed. Table 3.3 lists the metadata-related methods of the GetSensorData service.

Table 3.3 Metadata-related methods of the GetSensorData service

Method	Description
listMethods	List all methods provided in the service
getMethodOntology	Return the ontology of a specific method

The data-related methods provide the actual sensor data. These data come from sensor deployments, with each deployment having specific number and type of sensors. The data-related methods also include those methods that provide such information. This is critical for cross-disciplinary data sharing since a client VO that wishes to access the data from a particular server VO should first know *what* to ask for so that the server VO can provide the relevant data. Thus, the methods, such as listDeploymentSites, provide a list of all sensor deployment sites that the server VO wishes to expose to others. Having obtained the list of sites, the client VO can then choose one or more of the sites where it wishes to retrieve real-time or archived data.

Once the client VO has selected the deployment sites, the next step is to know what sensors have been deployed at that particular site, what are the attributes (such as data types, data availability time period, data format.) of each sensor and how to interpret the data and to translate it if necessary. For this purpose, the GetSensorData service provides a method getSiteOntology that encapsulates this information as an ontology in OWL format. The client can then process the OWL document to retrieve the details of the sensors and their data. Finally, the method getData can be invoked by the client with the specification of deployment sites, sensors, and period of data to be retrieved (if accessing archived data). The client can also choose from a number of formats in which the data should be provided, such as OWL, XML, and ASCII. Table 3.4 lists the data-related methods of the GetSensorData service.

3.4.3 Client-Side Service Implementation

The client VOs implement client-side services that talk to their server VO counterparts to obtain access to the resources. While different approaches can be adopted

Table 3.4 Data-related methods of the GetSensorData service

Method	Description
listDeploymentSites	Return a list of all sensor groups available
getSiteOntology	Return all attributes of the sensor group specified
getData	Return the data for the specified sensor group of the specified start

for client-side service implementation and deployment, the sequence in which various methods in the service are invoked will remain the same. Figure 3.11 shows a basic flow of method invocation. The client VO first initiates a service discovery request with parameters describing the high-level classification of the type of resource the client is seeking. At the broadest level, these parameters will include:

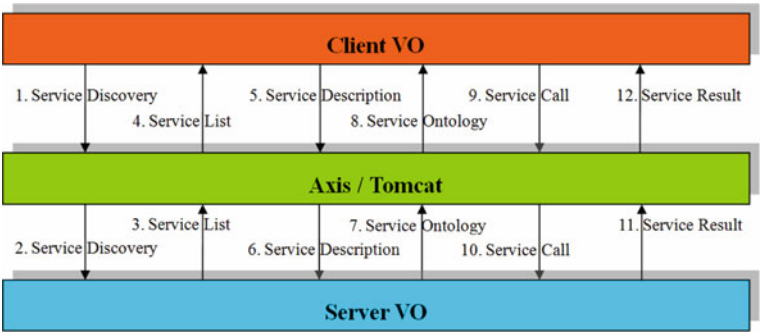


Fig. 3.11 Client VO-to-server VO service execution flow

1. Resource type (sensor, storage, compute, data, method)
2. Client credentials

While the resource-type parameters will be used by the server side to narrow down the type of services the client is looking for, the credentials will help in the client's authentication and the identification of his access level. Such service requests are received by the service containers such as Apache/Tomcat and handled by the services at the server VO. The services at the server side in turn provide a list of relevant services to the client. The key to this client–server communication is the exchange of OWL documents (such as services list, ontologies) that enable both client and server sides to accurately interpret the information supplied by the other party. The client VO selects a service from the list and requests the description of the service. The client VO at this stage expects to know how to invoke the service in order to gain access to a particular resource in the server VO (such as data, storage). The server VO responds with the ontology of the requested service in the OWL format describing service ontology, resource ontology, and resource access level. The client then invokes the service using the format prescribed in the ontology and waits for the results (data in case of a data request and access to compute/storage resource in case of a corresponding resource request).

We have implemented prototype client VO services in both Apache Axis and without Apache Axis scenarios. In case of Axis-based deployment, the following steps are needed to deploy the service and to invoke the sequence of methods in the service, as described in Fig. 3.11.

1. Request OWL/WSDL for the service from server VO.
2. Generate client-side stubs: ServiceLocator and SoapBindingStub.
3. Implement the client logic using the generated stubs.
4. Invoke the service.

For deploying a client on a non-Axis platform, the following steps are needed:

1. Discover the service locator (such as URI/URL) by requesting service advertisements from server VO.
2. Create a handle to the service.
3. Provide parameters needed by the service handle.
4. Invoke the service.

3.4.4 Fundamental and Composed Services

3.4.4.1 Users and Application Services

The users in the smart e-Science cyberinfrastructure usually access the services through Web-based interfaces (such as a Web portal). On the other hand, researchers usually use Web services to access the data as well as other resources exposed by different VOs. For users sending data acquisition requests through the Web or other TCP/IP networks, the *Grid User Job Submission (GUJSub)* service handles remote job submission. The job may make use of services such as data acquisition, weather statistics computation, alerts and notification, and visualization that have been implemented as part of the smart e-Science cyberinfrastructure. Otherwise, the user may submit his service to be executed. In such cases, the *Grid User Authentication (GUAuth)* service validates the user first before GUJSub submits the job for execution.

The Web interfaces also provide access to the *Workflow Creation* service that enables users to compose and submit complex computational and data acquisition tasks from basic services. Such tasks may include continuously acquiring sensor data from a particular geographical region, running statistical analysis on the data, and alerting the user in case some events are detected.

The *Users & Applications* (*UA*) components [34] also include *Data Access* application programming interfaces (APIs) that provide pull/push-based access to real-time as well as archived sensor data. The data are provided in various formats that can be parsed and processed, such as plain ASCII, XML, and OWL. These APIs are particularly useful for applications that require direct access to the data. Thus, apart from browsing the data manually via the Web portal, users can also access the data from their applications.

For example, in the case of the NWSP application, it should be noted that a general public user who is only interested in making use of weather-related services (such as obtaining a snapshot of current weather status, subscribing to weather

alerts) does not need to know the technical details of how the smart e-Science cyberinfrastructure has been implemented and how his request is serviced. Users who submit their own services to be executed on the smart e-Science cyberinfrastructure should know the application programming interfaces (APIs) as well as how the results of their jobs will be delivered to them. Finally, users who want to join the smart e-Science cyberinfrastructure as an independent VO need to know the details of publishing their resources and services to become shareable across the smart e-Science infrastructure.

3.4.4.2 Sensor Resources Proxy Services

The sensor networks are integrated with the grid infrastructure via a proxy. This proxy performs data acquisition and sensor network management tasks. Data acquisition tasks can be persistent or ad hoc. In persistent data acquisition, the proxy gathers data continuously from the sensor network, whereas in ad hoc data acquisition, the rate of data acquisition from the sensor network is not regular.

The proxy runs a *Grid Sensor Network Management* (*GSMan*) service that manages the sensor network to which the proxy connects. GSMan has components that perform sensor network administration and sensor data access. The administrative tasks include *Sensor Network Connectivity Management* to intelligently maintain connection with the sensor network *Sensor Profile Reporting* to provide meta information regarding the sensor nodes in the network and *Sensor Availability Tracking* to log information and provide alert on sensor nodes' health status. In terms of data acquisition, the *Data Acquisition Scheduling* component schedules the data sensing and reporting cycles for each of the sensor nodes in the network. This schedule is driven by the data acquisition requests submitted by users.

While GSMan handles the proxy's interface to the sensor network, the *UA-SR Interface Management* (*USIMan*) component manages the flow of requests and data between the components in the UA and the *Sensor Resources (SR)* layers. The *Grid User Authentication and Data Security* component of USIMan handles the security of the data as well as the sensor network that the user wants to access. The users in this category are usually collaborating researchers. USIMan carries out *Policy-based Data Acquisition* in conjunction with GSMan to obtain sensor data based on users' requests. In addition, it includes data processing components such as the *Data Cleaning* component that cleans the data to improve its quality and the *Data Archival and Conversion* component to log the data locally and convert it into suitable formats as needed by the users, such as XML.

Our current data representation format is XML based and it is similar to a simplified version of SensorML. Our format follows the standard XML marshaling, un-marshaling, and parsing standards. It defines groups of sensors bound together as weather stations followed by sequences of sensor data in the parameter–value pair format. The format is continuously being improved and extended as more heterogeneous sensors and more complicated processes are being integrated into the system.

3.4.4.3 Compute/Data Resources Services

The smart e-Science cyberinfrastructure provides middleware services to support heterogeneity in the infrastructure in terms of resources, services, and geographical location. These middleware services handle various important tasks as described below.

The *Resource Discovery and Brokerage* service provides seamless access to sensor networks and grid resources, irrespective of their specifications and location. It discovers sensor networks that can fulfill a user's (or application's) data requirements. It also discovers computational and storage resources needed by jobs submitted by users or hosted on the grid.

In order to do its job, the Resource Discovery and Brokerage service makes use of the *Meta Scheduling* service that maintains an ontology of resources and sensor networks in participating VOs. This ontology includes the description of sensor networks with the detailed capabilities and specifications of sensors, specifications of computational and data resources, and user access. Once the Resource Discovery and Brokerage service identifies the needed data and services, the meta scheduler helps to reserve and to provide access to particular resources to the jobs via Web Services Definition Language (WSDL).

For workflow-based jobs, the *Workflow Management* middleware service handles the execution of the user's job that may include executing multiple services on the grid and responding to the user with the results. The workflow management service makes use of the Resource Discovery and Brokerage service to discover the data as well as services that the user requires and then executes the job based on the execution policy defined by the user.

The smart e-Science cyberinfrastructure supports distributed framework development, i.e., the participating VOs may make the services running in their infrastructure to be accessible and used by others. Likewise, the computational as well as data resources in different VOs may also be published for smart e-Science *community* use. For this purpose, the smart e-Science middleware includes the *Resource Registration and Publishing* services for service registration, computational resource registration, storage resource registration, and their publishing. These services perform the task of registering the resources in smart e-Science ontology so that the Resource Discovery and Brokerage service can search them and make them available to users and other services when needed.

The *Job Management and Execution* service receives job requests from middleware and activates the services and executables identified by the grid meta scheduler. The *Load Balancing* service is utilized at this stage to identify a suitable grid node where the job should be executed. In the current implementation, the average CPU utilization of grid nodes is the parameter to be considered when selecting a node where a job should be dispatched.

For the purpose of failure detection and recovery, the *Monitoring and Fault Tolerance* services, based on tested schemes like heartbeats, are used in the grid infrastructures as well as sensor network proxies. The information collected includes dynamic statistics such as failure ratios and downtime/uptime ratios and

is used with the VO's ontology to provide recommendations as to the best possible resource usage plan. The fault tolerance component is used whenever a fault is detected to carry out proactive as well as reactive data transfer, job transfer, and service migration to ensure the seamless execution of a job.

For data resources, a *Semantic Data Conversion* service is provided that uses ontology-based metadata to make sense of the sensor data and help to archive it accurately. This service is critical since it enables seamless data sharing among VOs. There is no industry standard that defines how the data from sensor networks should be formatted and interpreted. Thus, this service converts sensor data from its raw format to the one that is tightly coupled with its semantics and archived as various ontology instances. These ontology instances are then provided to the users to allow them to interpret and process the data accurately.

3.4.5 Conceptual and Semantic Schema

The schema for smart e-Science was designed and implemented using a combination of MySQL and file system. MySQL is used to manage discrete sensor data, whereas raw binary data streams are co-managed using MySQL and the well-indexed and meta-tagged Linux file system. The ontology documents (OWL) for data and process ontology are also managed in the file system and indexed in the database. The resource registry is managed in the MySQL database. We are also experimenting with BLOB data type to handle raw binary data within the database and analyzing performance gains as compared to file-based binary data management. In order to support heterogeneous sensor types and extensibility to handle new types, the schema is not tied to any particular sensor type. Figure 3.12 shows the high-level database schema.

3.5 Conclusions

In this modern age of science, fundamental breakthroughs in any scientific field are only possible by undertaking scientific research that spans many scientific disciplines. e-Science promises to provide the backbone to support resource sharing among scientific communities. The provision of such an infrastructure is still in infancy and existing attempts comprise of incoherent efforts in grid computing, service-oriented architectures, and semantic Web. In this chapter, we have proposed a smart e-Science cyberinfrastructure that integrates existing technology in an innovative manner to realize the ultimate goal of sharing scientific resources. While building on top of grid, service-oriented architecture, and resource semantics, our proposed infrastructure builds an overarching ontology that binds every scientific resource in standards-based semantics and makes it shareable with the outside world via ontology-based services. We have also demonstrated the implementation of this smart e-Science cyberinfrastructure involving several large-scale scientific

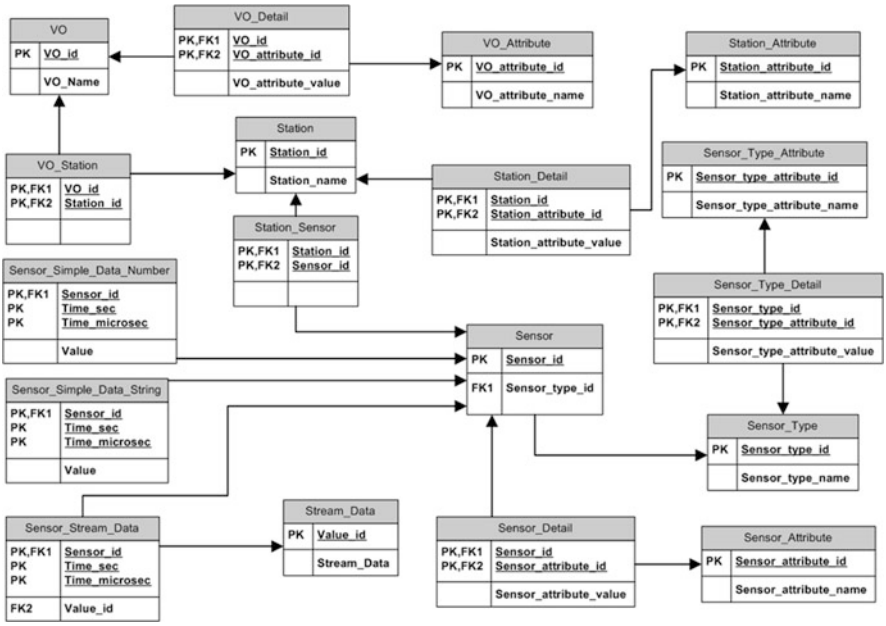


Fig. 3.12 High-level database schema

facilities that produce real-time sensor data, contain archived scientific data for their respective domains, and also contain computational and storage sources. We have shown that our infrastructure provides reusable services that seamlessly adapt to the respective domain based on its semantics. Thus, these services provide a means for composing more complex services in order to execute cross-domain scientific workflows.

Acknowledgments This work was supported by Microsoft Research, Intelligent Systems Center of Nanyang Technological University, Research Support Office of Nanyang Technological University, the Singapore National Research Foundation (NRF) under grant numbers NRF-G-CRP-2007-02 and NRF2008IDM-IDM001-005, and the Singapore National Research Foundation (NRF) through the Singapore-MIT Alliance for Research and Technology (SMART) Center for Environmental Sensing and Modeling (CENSAM).

References

1. Joseph, J., Fellenstein, C.: Grid Computing. IBM Press, USA (2004)
2. Sotomayor, B., Childers, L.: Globus Toolkit 4 Programming Java Services. Morgam Kaufmann, San Francisco (2006)
3. Leymann, F., Güntzel, K.: The business grid: Providing transactional business processes via grid services. In: Proceedings of the International Conference on Service Oriented Computing (ICSOC), Trento, Italy (2003)
4. McKee, P.: Grid – the 'white knight' for business? BT Technology Journal 23(3) (July 2005) 45–51
5. Zhang, L., Li, H., Lam, H.: Toward a business process grid for utility computing. In: IT Professional, IEEE Computer Society, 6(5), (September/October 2004) 62–64

6. Korpela, E., Werthimer, D., Anderson, D., Cobb, J., Lebofsky, M.: SETI@home – Massively distributed computing for SETI. Computing in Science & Engineering 3(January) (2001) 78–83
7. Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web Services: Concepts, Architectures and Applications. Springer, Berlin (2003)
8. Singh, M.P., Huhns, M.N.: Service-Oriented Computing Semantics, Processes, Agents. Wiley, New York (2005)
9. Stuckenschmidt, H.: van Harmelen, F.: Information sharing on the semantic web. Advanced Information and Knowledge Processing. Springer, Heidelberg (2005)
10. Uschold, M., Gruninger, M.: Ontologies: Principles, methods and applications. Knowledge Engineering Review 11(2) (1996) 93–115
11. Guarino, N., Carrara, M., Giaretta, P.: An ontology of meta-level categories. Proceedings of the 4th International Conference on Knowledge Representation and Reasoning (KR94), Morgan Kaufmann, San Mateo, CA (1994)
12. TeraGrid, http://www.teragrid.org/. Accessed 18 Jul 2010
13. Yamamoto, N., Nakamura, R., Yamamoto, H., Tsuchida, S., Kojima, I., Tanaka, Y., Sekiguchi, S.: GEO grid: Grid infrastructure for integration of huge satellite imagery and geosciences. In: Proceedings of the 6th IEEE/ACM International Conference on Computer and Information Technology (CIT), Seoul, Korea, (2006) 75
14. The London e-Science Center, http://www.lesc.ic.ac.uk. Accessed 18 Jul 2010
15. The Cambridge e-Science Center, http://www.escience.cam.ac.uk. Accessed 18 Jul 2010
16. Alper, P., Corcho, O., Kotsiopoulos, I., Missier, P., Bechhofer, S., Kuo, D., Goble, C.: S-OGSA as a reference architecture for OntoGrid and for the semantic grid. In: Proceedings of the 16th Global Grid Forum (GGF16) Semantic Grid Workshop, Athens, Greece, February (2006)
17. MyGrid: http://www.mygrid.org.uk/. Accessed 18 Jul 2010
18. Hull, D., Wolstencroft, K., Stevens, R., Goble, C., Pocock, M., Li, P., Oinn, T.: Taverna: A tool for building and running workflows of services. Nucleic Acids Research 34 (2006) 729–732
19. Cyberinfrastructure for the Centre for Environmental Sensing and Modeling, http://censam.mit.edu/research/res5/index.html#sec4. Accessed 18 Jul 2010
20. The National Weather Study Project, http://nwsp.ntu.edu.sg. Accessed 18 Jul 2010
21. Sensor Grid for GPS Data Processing Project, http://sensorgrid.ntu.edu.sg/gps. Accessed 18 Jul 2010
22. The LiveE! Project, http://www.live-e.org/en/index.html. Accessed 18 Jul 2010
23. e-Science Ontology, http://sensorgrid.ntu.edu.sg/SmarteScience.html. Accessed 18 Jul 2010
24. The Open Geospatial Consortium, http://www.opengeospatial.org. Accessed 18 Jul 2010
25. Semantic Web for Earth and Environmental Terminology (SWEET), http://sweet.jpl.nasa.gov/ontology. Accessed 18 Jul 2010
26. Geography Markup Language, http://www.opengeospatial.org/standards/gml. Accessed 18 Jul 2010
27. Sensor Web Enablement Working Group, http://www.opengeospatial.org/projects/groups/sensorweb. Accessed 18 Jul 2010
28. Sensor ML, http://www.opengeospatial.org/standards/sensorml. Accessed 18 Jul 2010
29. 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, Canada, July (2002)
30. Russomanno, D.J., Goodwin, J.C.: OntoSensor: An Ontology for Sensor Network Application Development, Deployment, and Management, Handbook of Wireless Mesh and Sensor Networking. McGraw Hill, New York (2008)
31. OWL Web Ontology Language Reference, Mike Dean and Guus Schreiber, Editors, W3C Recommendation, 10 February 2004, http://www.w3.org/TR/2004/REC-owl-ref-20040210/ Latest version available at http://www.w3.org/TR/owl-ref/. Accessed 18 Jul 2010

32. Liu, J., Zhao, F.: Towards semantic services for sensor-rich information systems. In: Proceedings of the 2nd IEEE/CreateNet International Workshop on Broadband Advanced Sensor Networks, Boston, MA, October (2005)
33. OWL-S. http://www.daml.org/services/owl-s. Accessed 18 Jul 2010
34. Lim, H.B., Iqbal, M., Wang, W., Yao, Y.: The national weather sensor grid: A large-scale cyber-sensor infrastructure for environmental monitoring. International Journal of Sensor Networks (IJSNet), Inderscience 7(1/2) (2010) 19–36

Chapter 4
Developing Ontologies within Decentralised Settings

Alexander Garcia, Kieran O'Neill, Leyla Jael Garcia, Phillip Lord, Robert Stevens, Oscar Corcho, and Frank Gibson

Abstract This chapter addresses two research questions: "How should a well-engineered methodology facilitate the development of ontologies within communities of practice?" and "What methodology should be used?" If ontologies are to be developed by communities then the ontology development life cycle should be better understood within this context. This chapter presents the Melting Point (MP), a proposed new methodology for developing ontologies within decentralised settings. It describes how MP was developed by taking best practices from other methodologies, provides details on recommended steps and recommended processes, and compares MP with alternatives. The methodology presented here is the product of direct first-hand experience and observation of biological communities of practice in which some of the authors have been involved. The Melting Point is a methodology engineered for decentralised communities of practice for which the designers of technology and the users may be the same group. As such, MP provides a potential foundation for the establishment of standard practices for ontology engineering.

4.1 Introduction

The maturity of a particular scientific discipline can be defined by its progress through three main stages. First, *innovation* followed by the subsequent dissemination of the resulting knowledge or artefact. Second, the formation of *communities* or collaborations, that utilise or build upon the innovations. Third, the proposal and agreement upon *standards* for protocols to achieve the unified and consistent progression of innovation and knowledge [1]. The discipline of ontology engineering can be thought of as progressing through the second stage of scientific maturity, moving from ontologies developed by a single authoritative expert to harvesting the collective intelligence of an application domain [2–4]. This trend is also reflected in

A. Garcia (✉)
University of Bremen, Bremen, Germany
e-mail: alexgarciac@gmail.com

H. Chen et al. (eds.), *Semantic e-Science*, Annals of Information Systems 11,
DOI 10.1007/978-1-4419-5908-9_4,

the availability of software supporting the engagement of several domain experts, communities, representing knowledge and developing ontologies [2, 5]. Therefore, ontology engineering is on the cusp of the third stage of scientific maturity, requiring the development of common working practices or standard methodologies.

Knowledge engineering (KE) is a field that involves integrating knowledge within computer systems [6] or the building, maintaining and development of knowledge-based systems [7]. Therefore, some of the methods proposed within the field of KE are applicable when building ontologies [8]. However, the experiences from KE have not always been applied when developing ontologies. In general KE methodologies focus primarily on the use of the ontology by a software system as opposed to the development of the ontology [9].

Within the domain of ontology engineering several methodologies have been proposed and applied [10–17]. The majority of the proposed methodologies have been engineered for centralised settings. However, none of these have gained widespread acceptance, predominant use or have been proven to be applicable for multiple application domains or development environments [18]. To date the community has not been widely involved or considered within ontology engineering methodologies. This situation has encouraged debate amongst those within the ontology community as to which methodology or combination of methodologies are most applicable [18, 9].

The language choice for encoding an ontology is still an open debate across the ontology building communities. This situation can be illustrated by the use of both the OBO format and the OWL within the bio-ontology community [19]. Conforming to or accepting a single formalism for ontology encoding would bring consistency and standardisation to the engineering methodology, such as tool support and reasoning engines. However, it is outside the scope of this work to recommend a particular formalism for ontology encoding. Therefore, the ontology methodologies are considered and analysed in a language-independent manner.

Whatever methodology emerges, it is essential that the methodology should be able to support the construction of ontologies by communities and utilise the collective intelligence of the application domain. Of the published methodologies that have been proposed or applied, no methodology completely satisfies all the criteria for collaborative development. To this end, we have reviewed the existing methodologies, identified commonalities and assessed their suitability for use within community ontology development. We have summarised these commonalities into a convergence of existing methodologies, with the addition of new aspects which we have termed the Melting Point (MP) methodology. The MP methodology builds upon the authors' experiences with community-developed ontologies, re-using methods and techniques that already exist and suggesting new mechanisms to cope with collaborative ontology development in decentralised settings.

4.1.1 Decentralised Communities

As knowledge is in a constant flux, ontologies should be flexible so they are reusable and easily extensible. This is not effortlessly achievable as representing

knowledge requires the active participation of domain experts. The majority of existing methodologies have been engineered for centralised settings, in which the ontology is developed and deployed on a one-off basis. Afterwards, the maintenance, as well as the evolution of the ontology, is left to the knowledge engineer and a reduced group of domain experts. This situation is also true throughout the development process: a reduced group of domain experts work together with the knowledge engineer to build the ontology. To date the community has not been widely involved or considered within ontology engineering methodologies.

The costs and efforts associated with the development of ontologies are considerable, it is therefore important to facilitate this process by allowing the community to participate in the development. By having this active participation some important aspects are covered: first, the quality of the model is constantly verified and second the evolution of the ontology is feasible. This paradigm follows the "wisdom of crowds" — assuming that more contributors implies higher quality or volume of information — as is employed within wiki-based collaborations. Collaboration is thus at the Melting Point in a methodology for developing ontologies.

4.1.2 Community-Driven Ontology Engineering

To illustrate the motivation and applicability of the MP methodology, examples are given from the life sciences, specifically the biomedical ontology domain. Within the knowledge-intensive biological domain, collaboration and community involvement is common place and encouraged in ontology development maintenance, evaluation and evolution.

Despite the lack of formal methodologies, bio-ontologies continue to be developed and the nature of this development has two very interesting properties. First, it is highly distributed; domain experts in any given sub-domain of the biological sciences are rarely in one place. Rather, they are distributed across the globe yet frequently interact to either collaborate or peer review each others' work. Hence, when biologists build ontologies, they tend to form virtual organisations in which experts with different but complementary skills collaborate in building an ontology for a specific purpose. The structure of this collaboration does not necessarily have a central control; different domain experts join and leave the network at any time and decide on the scope of their contribution to the joint effort. Leveraging this kind of virtual collaboration can be a powerful tool when constructing an ontology. Second, biological ontologies continue to evolve, even after the initial development drive. The continued evolution reflects the advancement of scientific knowledge discovery. New classes, properties and instances may be added at any time and new uses or extended scope for the ontology may be identified [17]. By engendering and facilitating this level of community participation, an ontology engineer can speed up the initial development and help to ensure that the ontology remains up to date as knowledge within the domain advances.

For example, the Ontology of Biomedical Investigations (OBI)[1] aims to provide an ontological representation of life science investigations covering common components of experimentation, such as equipment, materials and protocols. The developer community of OBI[2] is currently affiliated with 18 diverse biomedical communities, ranging from functional genomics to crop science to neuroscience. In addition to having a diverse community of expertise, the OBI developers work in a decentralised environment encompassing multiple countries and time zones.

The diversity of the life science domain results in a multitude of application domains for ontology development. To account for and identify available bio-ontologies the Open Biomedical Ontologies (OBO) Foundry [3] was formed. The OBO Foundry acts as a registry to collect public domain ontologies that, by design and revision, are developed by and available to the biomedical community, fostering information sharing and data interpretation. As of 23 October 2008 there are 76 registered ontologies at the OBO Foundry, representing knowledge domains ranging from Amphibian gross anatomy, infectious diseases to scientific experimentation. Although registered in the same library, the bio-ontologies, often present overlap in terminology or application domain. In addition to providing a registry, the OBO Foundry was formed to reduce ontology overlap and ensure bio-ontology orthogonality. Initial steps at achieving this aim have produced a set of design principles[3] to which domain ontologies should adhere, such as openness, a shared syntax and class definitions. However, the OBO Foundry does not suggest a community-orientated engineering methodology, methods or techniques by which these principles can be met.

Case studies have been described for the development of ontologies in diverse domains, yet surprisingly very few of these have been reported to have been applied to a domain allied to bioscience, the chemical ontology [13], and the ontology for immune epitopes [20] being noteworthy exceptions. The research focus for the bio-ontology community to date has typically centred on the development of domain-specific ontologies for particular applications, as opposed to the actual "how to" of building the ontology or the "materials and methods" [17, 21]. This has resulted in a proliferation of bio-ontologies, developed in different ways, often presenting overlap in terminology or application domain.

The biomedical domain is not the only domain where ontologies are being developed and applied. The Semantic Web (SW) encompasses a vision where knowledge and relationships between documents are disseminated via ontologies by annotating the current, largely human-accessible Web, to facilitate a Web amenable to machine processing [22]. Indeed, the creators of that vision consider the life sciences to potentially be the "incubator" community for the SW, as the physics community was for the Web [23]. Within the SW vision, as within the biomedical domain, the involvement of communities of practice is crucial, not only for the development, but also for the maintenance and evolution of ontologies.

[1] http://purl.obofoundry.org/obo/obi/

[2] http://obi-ontology.org/page/consortium

[3] http://www.obofoundry.org/crit.shtml

4.1.3 Upper Level Ontologies

As the biomedical domain is highly interconnected, domain ontologies may overlap with each other. For instance, OBI requires the availability of definitions for those chemicals used in any investigation. These definitions do not need to be developed within the OBI ontology as there is already a biomedical ontology for the domain of chemicals, called ChEBI [24]. Similarly, software making use of an ontology may require more than a single domain ontology. Typically, in these types of scenarios, it is necessary to integrate multiple ontologies into a single coherent narrative. In order to integrate or re-use specific domain ontologies following this "building-block" approach there has to be a high-level structure or common "scaffold" where different parts of different domain ontologies may be "plugged" into. To ensure ease of interoperation or re-use of a domain ontology, well designed and documented ontologies are essential and upper ontologies are fundamental in this integrative effort.

Upper level ontologies provide a domain-independent conceptual model that aims to be highly re-usable across specific domain applications. Most of the upper ontologies provide a general classification criterion that makes it easy to re-use, extend and maintain those existing ontologies required by a particular application. Therefore, it is essential, to aid interoperability and re-use, that ontology development methodologies should provide general guidelines for the use of upper level ontologies. These guidelines should cover the documentation of (i) the design decisions and the justification for choosing one upper ontology over another and (ii) examples that illustrate how they used in the conceptualisation of a particular domain. Examples of upper level ontologies include the Basic Formal Ontology (BFO) [25], DOLCE [26] and GFO [27]. This adoption has, however, not been documented within a methodological framework that facilitates both the adoption of the upper level ontology and its proper use. However, the OBO foundry has recommended that ontologies registered on the OBO Foundry should use BFO.

4.1.4 Dynamic Ontologies

Ontologies, like software, evolve over time; specifications often change as the development proceeds, making a straightforward path to the ontology unrealistic. Different software process models have been proposed; for instance, linear sequential models and prototyping models. Linear sequential models are also known as waterfall models [28, 29] and are designed for straight-line development. The linear sequential model suggests a systematic, sequential approach in which the complete system will be delivered once the linear sequence is completed [29]. The role of domain experts is passive as end-users of technology. They are placed in a reacting role in order to give feedback to designers about the product. The software or knowledge engineer leads the process and controls the interaction amongst domain experts. A high-speed adaptation of the linear sequential model is the

Rapid Application Development (RAD) model [30, 31]. This emphasises short development cycles for which it is possible to add new software components, as they are needed. RAD also strongly suggests reusing existing program components or creating reusable ones [29].

The prototyping model is more flexible as prototypes are constantly being built. Prototypes are built as a means for defining requirements [29], this allows for a more active role from domain experts. A quick design is often obtained in short periods of time. The model grows as prototypes are being released [32]; engineers and domain experts work on these quick designs. They focus on representational aspects of the ontology, while the main development of the ontology (building the models, defining what is important, documenting, etc.) is left to the knowledge engineer.

The evolutionary nature of the software is not considered in either of the aforementioned models, from the software engineering perspective evolutionary models are iterative, and allow engineers to develop increasingly more complex versions of the software [29, 31, 33]. Ontologies are, in this sense, not different from other software components for which process models have evolved from a "linear thinking" into evolutionary process models that recognise that uncertainty dominates most projects, that timelines are often impossibly short and that iteration provides the ability to deliver a partial but extendible solution, even when a complete product is not possible within the time allotted. Evolutionary models emphasise the need for incremental work products, risk analysis, planning followed by plan revision, and customer (domain expert) feedback [29].

4.1.5 The Melting Point: A Methodology for Distributed Community-Driven Ontology Engineering

A general purpose methodology should aim to provide ontology engineers with a sufficient perspective of the stages of the development process and the components of the ontology life cycle, and account for community development. In addition, detailed examples of use should be included for those stages, outcomes, deliverables, methods and techniques; all of which form part of the ontology life cycle [9, 34].

To address ontology development methodology in a distributed community environment the "The Melting Point" methodology is described. Consideration has been applied to the previously proposed methodologies and their integration adaptation and the re-use of components were possible within the MP. Several IEEE software engineering practices have also been included to formulate a methodology applicable to the requirements of community ontology development. The Melting Point also follows Sure's [9] work as it considers throughout the whole process the importance of the software applications that will ultimately use the ontology. The following sections not only present the MP methodology but also the relationship between methodological issues and the life cycle of community-based ontologies.

An analysis of current ontology engineering methodologies is presented in Section 4.2, emphasising the significance of commonalities across methodologies as

well as the engagement of communities of practice. Section 4.3 presents a detailed definition of the methodology and related components; methods, techniques, activities, and tasks of the MP methodology. Sections 4.4 and 4.5 contain discussion and conclusions, respectively.

4.2 Review of Current Methodologies

Melting Point espouses the combination of good practices from existing methodologies. A comparison of these methodologies is therefore appropriate, in order to give context to MP. Several ontology methodology approaches are analysed below, according to criteria described in detail in Section 4.2.1. These criteria are derived from the work done by Fernandez [35], Mirzaee [18] and Corcho et al. [36].

The engineering methodologies analysed are the Enterprise Methodology proposed by Uschold and King [10]; the TOVE Methodology proposed by Gruninger and Fox [11]; the Bernaras methodology proposed by Bernaras et al. [12]; the METHONTOLOGY methodology proposed by Fernandez et al. [37] the SENSUS methodology proposed by Swartout et al. [14]; the DILIGENT methodology proposed by Pinto et al. [15, 16]; the GM methodology proposed by Garcia et al. [17] the iCAPTURer Methodology proposed by Good et al. [38] and the NeOn methodology[4]. Table 4.1 provides a summary of the methodologies and the results of the analysis against the criteria. Complete details of the analysis have been provided in Appendix for reference.

4.2.1 Criteria for Review

C1. *Inheritance from knowledge engineering*. Ontology building is ultimately the assertion and representation of knowledge. Therefore, this criterion considers the influence traditional Knowledge Engineering (KE) has had on the methodologies studied.

C2. *Detail of the methodology*. This criterion is used to assess the clarity with which the methodology specifies the orchestration of methods and techniques.

C3. *Strategy for building the ontology*. This should provide information about the purpose of the ontology, as well as the availability of domain experts. There are three main strategic lines to consider: (i) the application of the ontology; (ii) the use and type of domain experts available and (iii) the type of ontology to be developed. These three aspects are defined in more detail from C3.1 to C3.3.

C3.1. *Application of the ontology*. This criterion describes how tightly coupled the ontology is going to be in relation to the application within

[4] http://www.neon-project.org

Table 4.1 Evaluation of ontology engineering methodologies according to common criteria (specified in Section 4.1). Full details are provided in Appendix A

	Inheritance from knowledge engineering	Detail of the metho-dology	Strategy for building the ontology			Strategy for identifying concepts	Recommended life cycle	Recommended methods, techniques and technology	Applicability	Community involvement	Knowledge elicitation
			Application of the ontology	Domain experts	Ontology type						
Enterprise	Partial	Very little	Application independent	N/A	N/A	Middle out	N/A	N/A	Enterprise ontology	N/A	N/A
TOVE	Small	Little	Application semi-independent	N/A	Domain and task ontologies	Middle out	N/A	N/A	Business and foundational ontologies	N/A	N/A
Bernaras	A lot	Very little	Application independent	N/A	Domain and task ontologies	Top down	Follows software	N/A	Electrical engineering domain	N/A	N/A
METH-ONTOLOGY	A lot	A lot	Application independent	N/A	In principle it is applicable to any kind of ontologies	Middle out	Evolutionary prototype	Some activities missing. Technology recommended	Multiple developments reported	N/A	Partially
SENSUS	Inexistent	Medium	Application semi-independent	N/A	In principle it is applicable to any kind of ontologies	N/A	N/A	N/A	Air campaign planning ontology	N/A	N/A
DILIGENT	Small	Small	Application dependent	N/A	N/A	Problem dependent	N/A	N/A	N/A	Yes	Partially
GM	Small	A lot	Application dependent	The focus is more on specialized domain experts	Domain and task ontologies	Top down	Evolutionary prototype	Some techniques and methods recommended Technology recommended	Developments reported for the bio domain	Yes	Supported
iCAPTURer	Little	A lot	Application-independent	Broad: All levels of domain experts considered	Domain and task ontologies	Bottom up	Evolutionary/ iterative	N/A	N/A	Yes	Integral
NeOn	Strongly influenced	A lot	Application-independent	Domain experts and ontology engineers assumed to be actively working together.	Domain and task ontologies	N/A	Recognised but not prescribed	Many, with specific detail	Multiple developments reported	Collaboration of the community is assumed	Significant

which, in principle, it should be used. This is often evaluated via competency questions. Competency questions can be typically classified in two forms: informal competency questions (natural language) and formal competency questions. The criteria for describing the level of application engagement are described as follows:

C3.1.1. *Application dependent*. The ontology is built on the basis of an application knowledge base, by means of a process of abstraction [35].
C3.1.2. *Application semi dependent*. Possible scenarios of ontology use are identified in the specification stage [35].
C3.1.3. *Application independent*. The process is totally independent of the uses to which the ontology will be put in knowledge-based systems or any software layer making use of the ontology.

C3.2. *Domain experts*. This criterion outlines the level of perceived expertise of the individuals consulted within the ontology development process. A domain expert can be graded in the following manner:

C3.2.1. *Specialised domain experts*. Those with an in-depth knowledge of their field. Within the biological context these are usually researches with vast laboratory experience, very focused and narrowed within the domain of knowledge. Having the specialised domain expert helps define very specific concepts within the ontology; this can lead to a strategy for identifying concepts known as the bottom-up approach (see C4.1).
C3.2.2. *Broader-knowledge domain experts*. Those who have a general knowledge or a higher level view of the domain(s). Having this kind of domain experts usually facilitates capturing concepts more related to high-level abstraction, and general processes, rather than specific vocabulary describing those processes. The ontology may be built from high-level abstractions downwards to specifics. This approach known as the top–down strategy for identifying concepts (see C4.2).

C3.3. *Ontology type*. The ontology-type criterion classifies the ontology being developed into the following types or categories [39, 40]:

C3.3.1. *Top-level ontologies*. These describe very general concepts like space, time, event, which are independent of a particular problem domain. Such unified top-level ontologies aim at serving large communities [9]. These ontologies are also known as foundational ontologies or commonly called upper ontologies; see, for instance, the DOLCE ontology [26].

C3.3.2. *Domain ontologies*. These are focused within a particular domain and describe specific vocabulary.
C3.3.3. *Task ontologies*. These describe vocabulary related to tasks, processes or activities.
C3.3.4. *Application ontologies*. As Sure [9] describes them, application ontologies are specialisations of domain and task ontologies as they form a base for implementing applications with a concrete domain and scope.

C4. *Strategy for identifying concepts*. There are three strategies regarding the construction of the ontology and the kinds of terms it is possible to capture [41]:

C4.1. *Bottom-up* work from the most concrete or specific concepts to the most abstract concepts. [41–43].
C4.2. *Top-down* work from the most abstract to the more domain/application specific concepts [35].
C4.3. *Middle-out* work from the most relevant to the most abstract and most domain/application-specific concepts [35, 40, 43].

C5. *Recommended life cycle*. This criterion evaluates whether and to what degree the methodology implicitly or explicitly proposes a life cycle [35].
C6. *Recommended methods and techniques*. This criterion evaluates whether or not there are explicit methods and techniques as part of the methodology. This is closely related to C2. An important issue to be considered is the availability of software supporting either the entire methodology or a particular method of the methodology. This criterion also deals with the methods or software tools available within the methodology for representing the ontology, for example, in OWL[5], or RDF[6].
C7. *Applicability*. As knowledge engineering is still in its infancy it is important to evaluate the methodology in the context of those ontologies for which it has been applied.
C8. *Community involvement*. As has been pointed out before in this chapter, it is important to know the level of involvement of the community. Phrasing this as a question, is the community a consumer of the ontology or is the community taking an active role in its development?
C9. *Knowledge elicitation*. Knowledge elicitation is a major bottleneck when representing knowledge [44]. It is therefore important to know if the methodology assumes knowledge elicitation to be an integral part of the methodology; does it describe the elicitation techniques?

[5] http://www.w3.org/TR/owl-ref/
[6] http://www.w3.org/RDF/

4.2.2 Finding the Melting Point

The considerable number of methodologies coupled with the limited descriptive detail of the ontology development approach makes it extremely difficult to present a consensus or a melting point where the methodologies converge. From the methodologies studied, very few clearly state the methods and techniques suggested in the methodology. However, the roles of those participating in the development of the ontology are clearly outlined. The following sections discuss the identified commonality and differences in approach, elucidated from the evaluation of the methodologies.

Common features. There are certain commonalities in the ontology development approach across the methodologies. For instance, all the studied methodologies consider an inception, formalisation as well as an evaluation phase. Figure 4.1 illustrates those shared stages across all methodologies. The DILIGENT and GM methodologies, however, present some fundamental differences when compared to the other methodologies — both were engineered for developing ontologies within geographically distributed settings. The differences identified between the DILIGENT and the GM methodology on the one hand and the other methodologies are presented and described below:

Life cycle. For both DILIGENT and GM, the ontology is constantly evolving, in a never-ending cycle. The life cycle of the ontology is understood as an open cycle in which the ontology evolves in a dynamic manner.

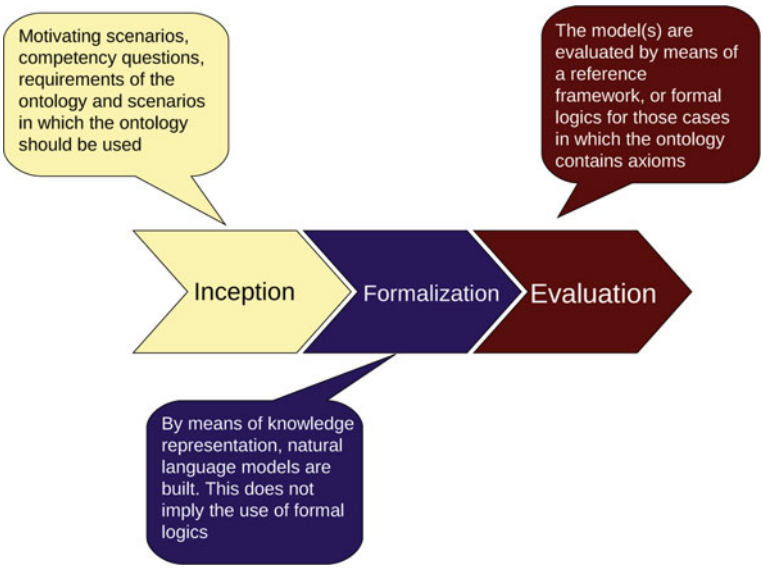


Fig. 4.1 Common features of the methodologies reviewed

Collaboration. For both DILIGENT and GM, a group of people agrees on the formal specification of the concepts, relations, attributes, and axioms that the ontology should provide. This approach empowers domain experts in a way that sets DILIGENT apart from the other methodologies.

Knowledge elicitation. Due in part to the involvement of the community and in part to the importance of the agreements, for DILIGENT and the GM methodology knowledge elicitation is assigned a high level of importance; it supports the process by which consensus is reached.

The GM, DILIGENT and NeOn methodologies consider the ontology to be constantly evolving. In fact, the life cycle spirals, with the ontology progressing over each iteration of the cycle. In addition, the GM methodology also emphasises the notion of collaboration in the development process, particularly during knowledge elicitation. The GM knowledge elicitation relies heavily on interaction; the higher the level of interaction amongst domain experts, the more refined the specific models are likely to be. Both DILIGENT and GM methodologies assume a leading role for the domain experts as well as a tight relationship between the ontology and the software application in which it will ultimately be used. Within the NeOn framework the focus is more on the network of ontologies rather than specifically on the act of collaboration amongst domain experts. However, NeOn offers a complete review of methods that in principle support collaboration when building ontologies; NeOn supports the collaboration over two main axes: argumentation and collaborative editing.

The GM, DILIGENT and NeOn methodologies consider the ontology to be constantly evolving. In fact, the life cycle spirals, with the ontology progressing over each iteration of the cycle. In addition, the GM methodology also emphasises the notion of collaboration in the development process, particularly during knowledge elicitation. The GM knowledge elicitation relies heavily on interaction; the higher the level of interaction amongst domain experts, the more refined the specific models are likely to be. Both DILIGENT and GM methodologies assume a leading role for the domain experts as well as a tight relationship between the ontology and the software application in which it will ultimately be used.

Differences amongst Methodologies. As illustrated by the summary table, no methodology completely satisfies all the criteria. Some of the methodologies, such as that of Bernaras, provide information about the importance of the relationship between the final application using the ontology and the process by which the ontology is engineered. This consideration is not always taken from the beginning of the development; clearly the kind of ontology that is being developed heavily influences this relationship. For instance, foundational ontologies rarely consider the software using the ontology as an important issue; these ontologies focus more on fundamental issues affecting the classification system such as time, space, and events. They tend to study the intrinsic nature of entities independently from the particular domain in which the ontology is going to be used [37].

The final application in which the ontology will be used also influences which domain experts should be considered for the development of the ontologies. For instance, specialised domain experts are necessary when developing application ontologies, domain ontologies or task ontologies, but they tend not to have such a predominant role when building foundational ontologies. For these kinds of ontologies philosophers and broader knowledge experts are usually more appropriate.

None of the methodologies investigated provided detail; the descriptions for the processes were scarce, and where present theoretical. There was no analysis of actual ontology building sessions. The methods employed during the development of the ontologies were not fully described. For instance the reasons for choosing a particular method over a similar one were not presented. Similarly there was no indication as to what software should be used to develop the ontologies. METHONTOLOGY was a particular case for which there is a software environment associated to the methodology; the recommended software WebODE [45] was developed by the same group to be used within the framework proposed by their methodology.

Although the methodologies investigated have different views on the life cycle of the ontology, only DILIGENT, NeOn and GM consider the life cycle to be dynamic. This is reflected in the processes these methodologies propose. The development happens in a continuum; some parts within the methodologies are iterative processes, but the steps are linear, taking place one after the other. In the case of DILIGENT the different view on the life cycle is clear. NeOn poses a view of the process that is closer to the one proposed by the MP methodology; it provides a clear view of the overall process and provides some detail as to the actual ontology building practice.

The lack of support for the continued involvement of domain experts who may be located around the world was not considered when engineering most of the studied methodologies. As both, the SW and the biodomain, pose a scenario for which information is highly decentralised, domain experts are geographically distributed and the interaction takes place mostly on a virtual basis, such consideration is important. For both cases the realisation of the SW vision, as well as the achievement of collaboration, is more about a change in people and communities of practices than it is about technology [4, 46].

Evolution and community involvement. Ontologies in the biomedical domain not only are domain and/or task specific but also application oriented. Within both, the SW and the biodomain, the construction of applications and ontologies will not always take place as part of the same software development projects. It is therefore important for these ontologies to be easily extensible; their life cycle is one in which the ontologies are in constant evolution, highly dynamic and highly re-usable. Ontologies in biology have always supported a wide range of applications; the microarray ontology (MO) [47], for instance, is used by several, unrelated microarray laboratories information systems around the world. In both scenarios, SW and biology, not only is the structure of the ontology constantly evolving but also the

role of the knowledge engineer is not that of a leader but more that of a facilitator of collaboration and communication amongst domain experts.

Parallels can be drawn between the biological domain and the SW. The SW-related scenarios are often described as being distributed, loosely controlled and evolving [15]. The main differences between the classic proposals for building ontologies and those requirements applied to the SW have been summarised by Pinto et al. [15], as well as Garcia et al. [17], and are described in four key points:

1. Distributed information processing with ontologies: Within the SW scenario, ontologies are developed by geographically distributed domain experts willing to collaborate, whereas KE deals with centrally developed ontologies.
2. Domain expert-centric design: Within the SW scenario, domain experts guide the effort while the knowledge engineer assists them. There is a clear and dynamic separation between the domain of knowledge and the operational domain. In contrast, traditional KE approaches relegate the role of the expert as an informant to the knowledge engineer.
3. Ontologies are in constant evolution in SW, whereas in KE scenarios, ontologies are simply developed and deployed.
4. Additionally, within the SW scenario, fine-grained guidance should be provided by the knowledge engineer to the domain experts.

Collaboration is present in the DILIGENT, iCAPTURer, NeOn and GM methodologies. However, neither DILIGENT nor GM propose methods for engaging the collaborators, nor do they provide clear methodological guidelines. Alternatively, NeOn proposes a set of methods and techniques for most of the steps described. Nevertheless, the process of knowledge elicitation, whether within the context of collaboration or as a focus group activity, is not fully addressed in most of the methodologies investigated. METHONTOLOGY and NeOn consider knowledge elicitation as part of the methodology, but there are no recommendations regarding knowledge elicitation methods.

One important feature that is not covered by any of the methodologies investigated is the use of upper level ontologies; as these are meant to support classification based on universal criterion it is important to understand the structure proposed by these ontologies in order to ease the integration of domain ontologies.

Collaboration, knowledge elicitation, a better understanding of the ontology life cycle and detailed description for the different steps involved are important criteria that should be documented to ensure that methodologies may be more efficiently replicated and applied. There is also an increasing need to emphasis the reuse of methodologies rather than developing ad hoc, de novo methodologies. The reuse of methodologies will go some way to ensure efficient development, interoperability and the elucidation of best practice in ontology development, irrespective of the domain of application. These are precisely the issues that a methodology for community-based ontologies needs to address.

4.3 The Melting Point Methodology

The following outlines the Melting Point methodology and can serve as a "manual" for ontology engineering. As mentioned in Section 4.1, many of the techniques are best practices chosen from other methodologies, and as such extensive reference is made back to these methodologies. The MP methodology aims to provide ontology developers with a detailed view of the processes that should take place when building ontologies; it supports the orchestration of steps in the development process based on the inputs consumed and outputs produced by the methods and techniques used. MP is not prescriptive about specific techniques or methods; ontology developers should consider the use of those that best suit their particular situation. This document, as well as several deliverables from the NeOn project are a good source of information regarding the methods and techniques available.

For the purpose of MP, the activities involved are framed within processes and activities, as illustrated in Fig. 4.3; this conception is promoted by METHONTOLOGY [35] for centralised settings. As these activities were not conceived within decentralised settings, their scope has been redefined, so that they better fit the life cycle of ontologies developed by communities. The methodology here presented emphasises: decentralised settings and community involvement. It also stresses the importance of the life cycle these ontologies follow, and provides activities, methods and techniques coherently embedded within this life cycle.

The methodology and the life cycle are illustrated in Fig. 4.2. The overall process starts with documentation and management processes; the development process immediately follows. Managerial activities happen throughout the whole life cycle; as the interaction amongst domain experts ensures not only the quality of the ontology, but also that those predefined control activities take place. The development process has five main activities: specification, conceptualisation, formalisation implementation and evaluation. Different prototypes or versions of the ontology are thus constantly being created. Initially these prototypes may be unstable, as the classes and properties may drastically change. In spite of this, the process evolves rapidly, achieving a stability that facilitates the use of the ontology; changes become more focused on the inclusion of classes and instances, rather than on the redefinition of the class hierarchy.

4.3.1 Definition of Terminology

To aid the clarity of the methodology descriptions the interpretation of key terminology must be made clear. The meaning of the terms methodologies, techniques and methods follow the Institute of Electrical and Electronics Engineers (IEEE) descriptions [18, 19], as recommended by Perez-Gomez et al. [32], [48], Fernandez et al. [37] Pinto et al. [15, 49] and Garcia et al. [17]. Both Fernandez et al. and Perez-Gomez et al. emphasise the importance of complying with the Institute of Electrical and Electronics Engineers (IEEE) standards, more

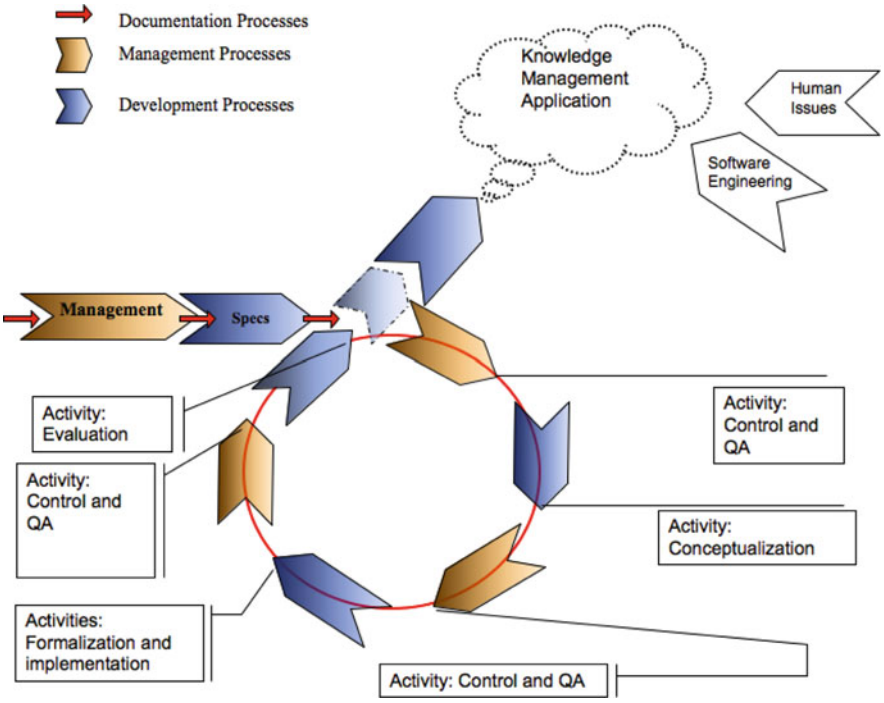


Fig. 4.2 Life cycle, processes, activities and view of the methodology

specifically with the IEEE standard for software quality assurance plans [50]. Not only does standards compliance ensure careful and systematic planning for the development, but it also ensures the applicability of the methodology to a broad range of problems. As such, we have also adopted terminology from the above-mentioned IEEE standard. A *methodology* should be interpreted as a "comprehensive integrated series of techniques or methods creating a general system theory of how a class of thought-intensive work ought to be performed" [18]. Methodologies are composed of both techniques and methods. A *method* is an "orderly" process or procedure used in the engineering of a product or performing a service [20]. A *technique* is a "technical and managerial procedure used to achieve a given objective" [18]. Thus methodologies bring together techniques and methods in an orchestrated way such that the work can be done. Figure 4.3 illustrates these relationships graphically.

Greenwood [51] and Gomez-Perez et al. [52] present these terminological relationships in a simple way: "a method is a general procedure while a technique is the specific application of a method and the way in which the method is executed" [52]. According to the IEEE [53] a process is a "function that must be performed in the software life cycle. A process is composed by activities". The same set of standards defines an activity as "a constituent task of a process" [53]. A task is the atomic unit

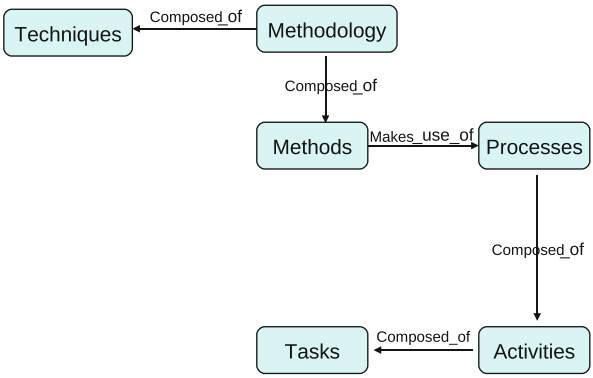


Fig. 4.3 Relationships amongst ontology engineering terms

of work that may be monitored, evaluated and/or measured; more formally, a task is "a well defined work assignment for one or more project member. Related tasks are usually grouped to form activities" [53].

4.3.2 Management Processes

The management activities are initiated as soon as there is a motivation (specification) and a decision for developing the ontology, therefore an artefact and a process to manage. The management process continues through the remainder of the ontology development process. Some of the activities involved in the management processes are

Scheduling. Scheduling identifies tasks, time and resources needed.
Control. Control ensures that the planned tasks are completed.
Inbound interaction. Inbound interaction specifies how the interaction amongst domain experts will take place, for instance, by phone calls, mailing lists, wiki and static Web pages.
Outbound interaction. As different communities should in principle be allowed to participate, there has to be an inclusion policy that specifies how a new community could collaborate and engage with the ongoing development.
Quality assurance. This activity defines minimal standards for the outputs from each and every process, activity or task carried out during the development of the ontology.

Scheduling project management techniques can be employed such as Gantt charts to and define milestones and deadlines. Several software suites exist to assist in project management, both commercial and open source. Specific technologies for documentation and communication are discussed in Garcia et al.17]. In addition,

more generic content management and communication systems can be employed for documenting and communicating the management process, such as those identified and reviewed by Mooney and Baenziger [54]. For both scheduling and controlling, the software tool(s) should in principle

– help to plan the activities and tasks that need to be completed,
– give a basis for scheduling when these tasks will be carried out,
– facilitate planning the allocation of resources needed to complete the project,
– help to work out the critical path for a project where one must complete it by a particular date,
– facilitate the interaction amongst participants and
– provide participants with simple means for exchanging information.

4.3.3 Documentation Processes

The documentation is a continuum process throughout the entire development of the ontology. This documentation should make it possible for new communities of practice to get involved in the development of the ontology. These include early processes such as the specification of the purpose of the ontology right through to later processes such as formalisation and evaluation of the ontology.

Documenting classes and properties. Although documentation can happen naturally, facilitated by discussions via an email basis, it is often difficult to follow the argumentative thread. Even so, the information contained in mailing lists is useful and should whenever possible be related to classes and properties. Use cases, in the form of examples for which the use of a term is well illustrated, should also be part of the documentation of classes and properties. Ontology editors allow domain experts to comment on the ontology; this kind of documentation is useful, as it reflects the understanding of the domain expert. For classes and properties there are three main sources of documentation:

Mailing lists. Discussions about why a class should be part of the ontology, why it should be part of a particular class hierarchy, how it is being used by the community, how a property relates two classes, and in general all discussions relevant to the ontology happen on an email basis.

On-the-ontology comments. In the cases when domain experts are familiarised with the ontology editor, they usually comment on classes and properties.

Use cases. This should be the main source of structured documentation provided by domain experts. However, gathering use cases is often difficult and time consuming. The use cases should illustrate how a term is being used in a particular context, how the term is related to other terms, and those different uses or meanings a term may have. Guidance is available for the construction of use cases when developing software; however, this direction is not available when building ontologies. From those experiences in which

the author participated some general guide can be drawn, for instance, use cases should be brief, they should be based upon real-life examples, knowledge engineers have to be familiar with the terminology as well as with the domain of knowledge because use cases are usually provided in the form of narratives describing processes, graphical illustrations should be part of the use case, and also whenever possible concept maps, or other related KA artefacts, should be used.

4.3.4 Development-Oriented Processes

These are the processes by which the ontology is actually built and represent the core of the methodology. The life cycle, documentation and management provide a framework in which development-oriented processes are embedded.

Specification. The specification of the ontology involves defining the motivation; in other words why the development of an ontology is required for the application domain. The specification phase can also be called a feasibility study and includes addressing straightforward questions such as "What is the ontology going to be used for?", "How is the ontology ultimately going to be used by the software implementation?", "What do we want the ontology to be aware of?" and "What is the scope of the knowledge we want to have in the ontology?". The answers to these questions are typically represented as competency questions, which define the requirements of the ontology. The requirements are dependent on the motivation and are described as informal questions or tasks that an ontology must be able to answer or perform. In other words, competency questions are those questions for which we want the ontology to be able to provide support for reasoning and inferring processes [17]. It is often helpful to include competency questions, as they can help to enforce the boundaries of the scope of the ontology.

Conceptualisation. The conceptualisation of the ontology is the process of identifying the key concepts that exist in the domain, their properties and the relationships that hold between them; this includes identifying natural language terms to refer to such concepts, relations and attributes as well as structuring domain knowledge into explicit conceptual models [55]. Gruber's design principles. [4] are relevant to the conceptualisation process as described below:

Gruber's first principle. "The conceptualisation should be specified at the knowledge level without depending on a particular symbol-level encoding."
Gruber's second principle. "Since ontological commitment is based on the consistent use of the vocabulary, ontological commitment can be minimised by specifying the weakest theory and defining only those terms that are essential to the communication of knowledge consistent with the theory."
Gruber's third principle. "An ontology should communicate effectively the intended meaning of defined terms. Definitions should be objective. Definitions can be stated on formal axioms, and a complete definition

(defined by necessary and sufficient conditions) is preferred over a partial definition. All definitions should be documented with natural language."

The process of conceptualisation typically involves the activities of domain analysis (DA) and knowledge elicitation (KE) and knowledge acquisition (KA). DA is the process by which a domain of knowledge is analysed in order to find common and variable components that best describe that domain. KE the process of collecting from a human source of knowledge, information that is relevant to that knowledge [44]. KA includes the elicitation, collection, analysis, modelling and validation of knowledge for knowledge engineering and knowledge management projects. The notion for both KA and KE comes from the development of knowledge bases; for the purposes of developing ontologies, KA and KE can be considered as transposable terms. KA and DA are interchangeable and complementary activities by which the information used in a particular domain is identified, captured and organised for the purpose of making it available in an ontology [56].

Those activities related to DA and KA focus more on capturing and representing knowledge in a more immediate manner and not necessarily on having logical expressions as part of the models; whereas when formalising and evaluating an ontology, activities and tasks are more oriented to include logical constrains and expressions. DA and KA may be seen as the art of questioning, since ultimately all relevant knowledge is either directly or indirectly in the heads of domain experts. This activity involves the definition of the terminology, i.e. the linguistic phase. This starts by the identification of those reusable ontologies and terminates with the baseline ontology, i.e. a draft version containing few but seminal elements of an ontology.

Identifying available sources of knowledge is also important; by doing so it can help to refine or confirm the ontology specification. In the bio-ontology domain this process can be facilitated by the OBO Foundry, which is a registry of available and accessible domain ontologies. Searching the registry can be made possible via the BioPortal from the National Center for Biomedical Ontology (NCBO) [57] or the Ontology Lookup Service[7]. The OLS provides a user-friendly single entry point for querying publicly available ontologies in the Open Biomedical Ontology (OBO) format. By means of the OLS it is possible to verify if an ontology term has already been defined and in which ontology is available [58].

The following criteria are important during knowledge acquisition [17]:

Accuracy in the definition of terms. The linguistic part of the ontology development is also meant to support the sharing of information/knowledge. The availability of context as part of the definition is useful when sharing knowledge.

Coherence. The narrative should be coherent; descriptions should make sense within the context in which they are intended to have a meaning.

[7] http://www.ebi.ac.uk/ontology-lookup/

Moreover, narratives should provide examples from which instances can be gathered.

Extensibility. This approach may be seen as an aggregation problem; CMs are constantly gaining information, which is always part of a bigger narration. Extending the conceptual model is not only about adding more detail to the existing CMs or just about generating new CMs; it is also about grouping concepts into higher-level abstractions and validating these with domain experts. Scaling the models involves the participation of both the domain experts and the knowledge engineer. It is mostly done by direct interview and confrontation with the models from different perspectives. The participation of new "fresh" domain experts, as well as the intervention of experts from allied domains, allows analysing the models from different angles. This participatory process allows re-factorising the models by increasing the level of abstraction.

The OBO Foundry has tried to define their criteria for defining terms. These OBO Foundry naming conventions[8] outline how to represent class labels and definitions to maintain consistency within one ontology and to provide a common naming conventions for integration across resources to avoid conflicts both at a human readable level and at a logical level.

For the purpose of DA and KA it is critical to elicit and represent knowledge from domain experts. They do not, however, have to be aware of knowledge representation languages; this makes it important that the elicited knowledge is represented in a language-independent manner. Researchers participating in knowledge elicitation sessions are not always aware of the importance of the session; however, they are aware of their own operational knowledge. This is consistent with the first of Gruber's design principles.

Regardless of the syntactic format in which the information is encoded domain experts have to communicate and exchange information. For this matter it is usually the case that wide general theories, principles, broad-scope problem specifications are more useful when engaging domain experts in discussions, as these tend to contain only essential basic terms, known across the community and causing the minimal number of discrepancies (see the second design principle). As the community engages in the development process and the ontology grows, it becomes more important to have definitions that are usable by computer systems and humans (see the third design principle). The relevant milestones, techniques and tasks for DA- and KA-related activities are

Tasks. Focal groups, limited information and constrained-processing tasks, protocol analysis, direct one-to-one interviews, terminology extraction, and inspection of existing ontologies.

[8] http://www.obofoundry.org/wiki/index.php/Naming

Techniques. Concept mapping, sorting techniques, automatic or semi-automatic terminology extraction, informal modelling and identifying pre-existing resources.

Milestones. Baseline ontology, knowledge sources, basic terminology, reusable ontologies.

Formalisation. Formalisation of the ontology is the activity during which the classes are constrained and instances are annotated against their corresponding classes. For example, "a male is constrained to be an animal with a y-chromosome". During the formalisation domain experts and knowledge engineers work with an ontology editor. When building iterative models and formalising the ontology the model grows in complexity; instances, classes and properties are added and logical expressions are built in order to have definitions with necessary and sufficient conditions. For both formalisation and iterative building of models, Gruber's fourth designing principle and Noy and McGuinness' guidelines [59] are applicable:

Gruber's fourth principle. "An ontology should be coherent: that is, it should sanction inferences that are consistent with the definitions. [. . .] If a sentence that can be inferred from the axioms contradicts a definition or example given informally, then the ontology is inconsistent."

Noy and McGuinness' first guideline. "The ontology should not contain all the possible information about the domain: you do not need to specialise (or generalise) more than you need for your application."

Noy and McGuinness' second guideline. "Subconcepts of a concept usually (i) have additional relations that the superconcept does not have or (ii) restrictions different from these of superconcepts or (iii) participate indifferent relationships than superconcepts. In other words, we introduce a new concept in the hierarchy usually only when there is something that we can say about this concept that we cannot say about the superconcept. As an exception, concepts in terminological hierarchies do not have to introduce new relations."

Noy and McGuinness' third guideline. "If a distinction is important in the domain and we think of the objects with different values for the distinction as different kinds of objects, then we should create a new concept for the distinction."

Implementation. The implementation of the ontology concerns the choice and justification of the encoding formalism, for example, the OBO format or the Web Ontology Language (OWL). The choice and the justification of a language take into account the required expressivity demanded by the specification process and by extension the tools required to facilitate the encoding. For example, if the chosen language was OWL, then it would be appropriate to use an ontology editor such as Protege[9]. Ultimately implementation is concerned with encoding the decisions

[9] http://protege.stanford.edu/

made as part of the formalisation process. However, the implementation process and the formalisation process can often happen simultaneously as an iterative process.

Iterative building of ontology models (IBOM). Iterative building of informal ontology models helps to expand the glossary of terms, relations, their definition or meaning, and additional information such as examples to clarify the meaning where appropriate. Different models are built and validated with the domain experts. There is a fine boundary between the baseline ontology and the refined ontology; both are works in progress, but the community involved has agreed upon the refined ontology.

Methods, techniques and milestones for the IBOM. Some milestones, techniques and tasks for IBOM related activities are

Tasks. Focal groups.
Techniques. Concept mapping, informal modelling with an ontology editor.
Milestones. Refined ontology.

4.3.5 Evaluation

There is no unified framework to evaluate ontologies and this remains an active field of research [32]. When developing ontologies on a community basis four main evaluation activities have been identified:

Specification evaluation. The specification defines the motivation and the scope of the ontology in the form of competency questions. Specification evaluation concerns the ability of the ontology to answer the competency questions and therefore demonstrate fulfilment of the intended scope.

Application-dependent evaluation. It is considered that ontologies should be evaluated according to their fitness for purpose, i.e. an ontology developed for annotation purposes should be evaluated by the quality of the annotation and the usability of the annotation software [17]. The community carries out this type of evaluation in an interactive manner; as the ontology is being used for several purposes a constant feedback is generated. The feedback thus gathered also helps in the evolution of the ontology; as the community comments on an ontology term being used to annotate a resource, ontology engineers are able to include, delete or edit terms in the ontology. This makes it possible for the community to effectively guarantee the usability and the quality of the ontology. By the same token, the recall and precision of the data, and the usability of the conceptual query builder, should form the basis of the evaluation of an ontology designed to enable data retrieval.

Terminology evaluation. This activity was proposed by Perez-Gomez et al. [60]. The goal of the evaluation is to determine what the ontology defines and how accurate these definitions are. Perez-Gomez et al. provides the following criteria for the evaluation:

Consistency. It is assumed that a given definition is consistent if, and only if, no contradictory knowledge may be inferred from other definitions and axioms in the ontology.

Completeness. It is assumed that ontologies are in principle incomplete [32, 60], however, it should be possible to evaluate the completeness within the context in which the ontology will be used. An ontology is complete if and only if *All that is supposed to be in the ontology is explicitly stated, or can be inferred*.

Conciseness. An ontology is concise if it does not store unnecessary knowledge, and the redundancy in the set of definitions has been properly removed.

Taxonomy evaluation. This evaluation is usually carried out by means of reasoned systems such as RACER [61] and Pellet [62]. The knowledge engineer checks for inconsistencies in the taxonomy, these may due to errors in the logical expressions that are part of the axioms.

4.4 Discussion

4.4.1 Melting Point Evaluated

The Melting Point (MP) methodology emphasises an integral knowledge management cycle. It is influenced by METHONTOLOGY and the work done by Sure in the field of knowledge management. The MP makes use of several methods and techniques, defining the steps which should be undertaken. The MP methodology stresses the importance of the orchestration of methods and techniques based on coherence between outcomes and deliverables for each step, thus proposing a flexible structure that can be adapted without losing rigor in the process. When evaluated against the criteria presented in Section 4.1, the MP methodology can be seen to have the following properties:

C1. *Inheritance from knowledge engineering*. Highly influenced by knowledge engineering.
C2. *Detail of the methodology*. Although it defines steps the MP methodology stresses the importance of an orchestration based on those outcomes and deliverables from each step. The MP aims for a flexible rigor, rather than a strict series of steps.
C3. *Strategy for building the ontology*.

C3.1. *Application of the ontology*. application independent.
C3.2. *Domain experts*. The methodology is intended to make use of knowledge gathered from all levels of domain experts. It is assumed an active participation of domain experts.
C3.3. *Ontology type*. The methodology is best suited for domain ontologies.

C4. *Strategy for identifying concepts*. Concepts are identified by a variety of methods and techniques; the MP does not enforce the use of a particular method or technique; it proposes processes for which there can be several methods and techniques available. It assumes an active participation of domain experts in that for the MP methodology domain experts are also modelers.
C5. *Recommended life cycle*. Processes, activities and tasks are proposed and orchestrated within an incremental evolutionary spiral model.
C6. Recommended methods and techniques.] The MP methodology proposes some methods and techniques; these are, however, changeable as the methodology does not emphasise the use of particular methods and techniques but rather stresses the impotence of an orchestrated Knowledge management process.
C7. *Applicability*. Parts of the proposed methodology have been applied and reported [17, 63]. The MP methodology is based upon these experiences and on the observation of several ontology development processes such as the CARMEN project[10].
C8. *Community involvment*. Active steps are taken to ensure that the community takes a leading role in the development process.
C9. *Knowledge elicitation*. Knowledge elicitation is an integral part of the overall process.

4.4.2 IEEE Standards Compliance

As discussed by [34], METHONTOLOGY is the only methodology that rigorously complies with IEEE standards; this facilitates the applicability and extendibility of the methodology. Other methodologies, such as those studied by [42] do not intentionally meet the terms posed by the IEEE. However, some of the proposed activities by those ontologies may be framed within IEEE standards. The Melting Point methodology proposed here reuses and adapts many components from METHONTOLOGY and other methodologies within the context of decentralised settings and participatory design. It also follows Sure's [9] work as it considers throughout the whole process the importance of the software applications that will ultimately use the ontology. The work done by Sure is complementary to the one presented in this chapter, as both works study different edges of the same process: developing knowledge-based software.

4.4.3 Quality Assurance

METHONTOLOGY allows for a controlled development and evolution of the ontology placing special emphasis on quality assurance (QA) thought the processes.

[10] http://carmen.org.uk/

Although QA is considered, the authors do not propose any methods for this specific task. Management, development and support activities are carried out in a centralised manner; a limited group of domain experts interact with the knowledge engineer, conceptualise and prototype the ontology, successive prototypes are then built, the ontology gains more formality (e.g. logical constraints are introduced) until it is decided that the ontology may be deployed. Once the ontology has been deployed a maintenance process takes place. Neither the development nor the evolution of the ontology involves a decentralised community; the process does not assume a constant incremental growth of the ontology as it has been observed, and reported by [17] QA is also considered to be a centralised activity, contrasting with the way decentralised ontologies promote the participation of the community in part to ensure the quality of the delivered ontology.

4.4.4 Activities Become Interrelated

As those required ontologies grow in complexity so does the process by which they are obtained. Methods, techniques, activities and tasks become more group-oriented, making it necessary to re-evaluate the whole process as well as the way by which it is described. The IEEE proposes a set of concepts that should in principle facilitate the description of a methodology; however, these guidelines should be better scoped for decentralised environments.

Activities within decentralised ontology developments are highly interrelated. However, the maturity of the product allows engineers and domain experts to determine boundaries and by doing so establishing milestones for each and every activity and task. Although managerial activities are interrelated and impact at a high level those development processes it is advisable not to have rigid management structures. For instance, control and inbound–outbound activities usually coexist with some development activities when a new term needs to be added. This interaction requires the orchestration of all the activities to ensure the evolution of the ontology. This interaction and orchestration of activities with defined boundaries and milestones are evident in the bio-ontology domain from the development of the Proteomics Standards Initiatives (PSI) sample processing and separations controlled vocabulary, sepCV. The PSI aims to facilitate global proteomics models for publication, data storage,data comparisons and dataintegration and to standardise and advance proteomics research [64]. To this end, they have developed minimum reporting guidelines [65], data transfer formats and ontologies to control the terminology used for reporting. The sepCV ontology had an initial specification and therefore milestones to represent the technology of gel electrophoresis [66]. However, its scope was then expanded to cover gel image informatics, so the life cycle continued collecting and representing community-defined concepts for both gel electrophoresis and gel informatics. In addition to these two technologies the sepCV is also expected to expand its specification to cover other separation technologies, such as column chromatography and capillary electrophoresis, with the

consequences that these interactions require the orchestration of all the activities to ensure the evolution of the ontology to fit dynamic boundaries and expanding specification over its life cycle.

4.4.5 Recommended Life Cycle: Incremental Evolutionary Spiral

When communities are developing ontologies the life cycle varies. The ontology is not deployed on a one-off basis; there is thus no definitive final version of the ontology. The involvement of the community allows for rapid evolution, as well as for very high-quality standards; errors are identified and discussed then corrections are made available within short time frames.

The model upon which this proposed methodology is based brings together ideas from, linear sequential modelling [29, 67], prototyping, spiral [68], incremental [69, 70] and the evolutionary models [29, 71]. Due to the dynamic nature of the interaction when developing ontologies on a community basis the model grows rapidly and continuously. As this happens prototypes are being delivered, documentation is constantly being generated, and evaluation takes place at all times as the growth of the model is due to the argumentation amongst domain experts. The development process is incremental as new activities may happen without disrupting the evolution of the collaboration. The model is therefore an incremental evolutionary spiral in which tasks and activities can coexist simultaneously at some level of detail. As the process moves forward activities and/or tasks are applied recursively depending on the needs. The evolution of the model is dynamic and the interaction amongst domain experts and with the model happens all the time. Figure 4.4 illustrates the model as well as how processes, activities and tasks are consistent with the model.

4.5 Conclusions

The Melting Point methodology proposed here reuses some components that various authors have identified as part of their methodologies for ontology development. This chapter has investigated how to use these components within decentralised settings, using the biomedical domain as an example. A domain where community development is critical to understanding a large, complex and an ever expanding knowledge base. Not only can the Melting Point methodology be demonstrated in the life science domain, the methodology can also be applicable to the development of the knowledge infrastructure of the Semantic Web, a decentralised environment by design.

The Melting Point methodology stresses the importance of a detailed description of the methods, techniques, activities, and tasks that could be used for developing community-based ontologies. Furthermore, a discussion of how the development process evolves adapts and expands with increasing or redefining of the ontology specification is presented within the life cycle model of these ontologies.

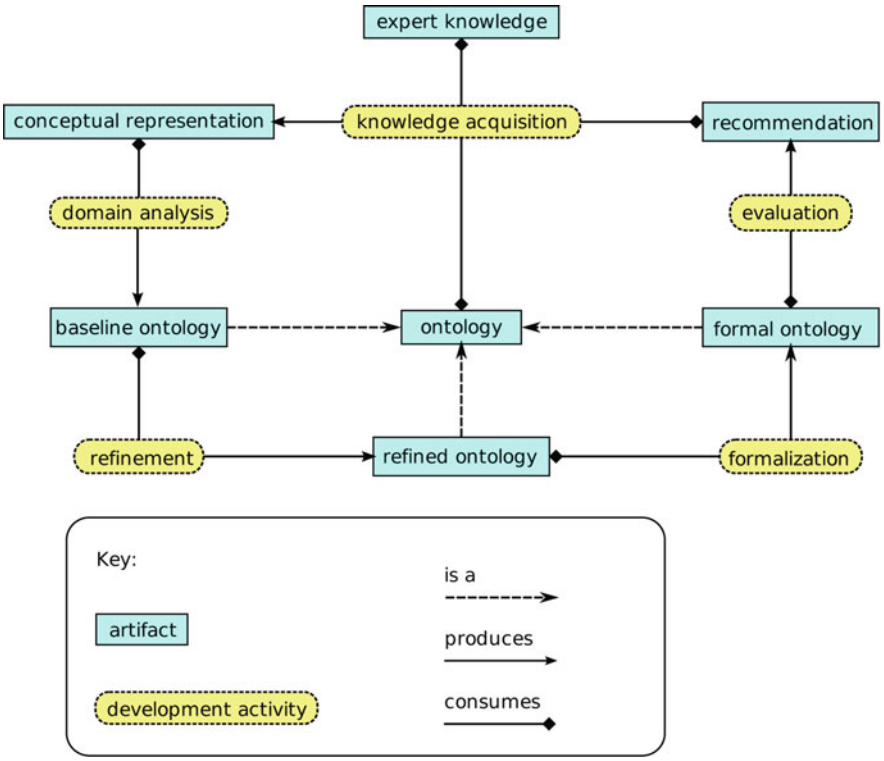


Fig. 4.4 A view of the whole process, showing development activities and the artefacts they produce and consume

The adoption of the Melting Point methodology should provide a level of rigour and consistent development of ontologies, with a particular focus on community contribution. The methodology may facilitate a process by which the OBO Foundry principles for bio-ontology development[11] can be achieved. Ontologies developed within the same methodology framework may aid in increasing ontology interoperability and integration, as the processes and design decisions can be disseminated upon publication and therefore followed and evaluated.

As we increasingly build large ontologies against complex domain knowledge in a community and collaborative manner there is an identified need for a methodology to provide a framework for this process. A shared methodology tailored for the decentralised development environment, facilitated by the Internet should increasingly enable and encourage the development of ontologies fit for purpose. The Melting Point methodology provides this framework which should enable the ontology community to cope with the escalating demands for scalability and

[11] http://www.obofoundry.org/crit.shtml

repeatability in the representation of community-defined knowledge bases, such as those in biomedicine and the Semantic Web.

A. Appendix: Review of Methodologies

A.1 The Enterprise Methodology

Uschold and King proposed a set of four activities that are listed here and illustrated in Fig. 4.5

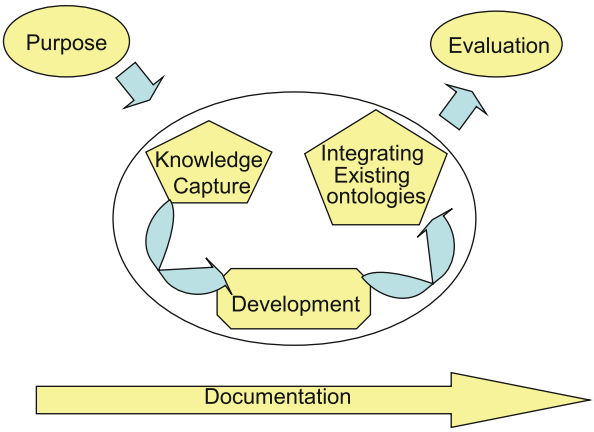


Fig. 4.5 Uschold and King methodology

1. Identify the purpose and scope of the ontology
2. Build the ontology, for which they specify three activities:
   - Knowledge capture
   - Development / coding
   - Integrating with other ontologies
3. Evaluate
4. Document the ontology

C1. The methodology does not explicitly inherit methods from knowledge engineering. Although Uschold and King identify steps that are in principle related to some methodologies from knowledge engineering. Neither a feasibility study nor a prototype method is proposed.

C2. Stages are identified, but no detail is provided. In particular the Ontology Coding Integration and Evaluation sections are presented in a superfluous manner [18].

C3. Limited information is provided. The proposed method is application independent and very general, in principle it is applicable to other domains. The authors do not present information about the kind of domain experts they advise working with.
C4. Uschold and Kind do not provide a clear criterion for the selection of either approach. For Uschold and King the disadvantage of using the top-down approach is that by starting with a few general concepts there may be some ambiguity in the final product. Alternatively, with the bottom-up approach too much detail may be provided and not all this detail could be used in the final version of the ontology [41]. This in principle favours the middle-out approach proposed by Lakoff [43]. The middle-out is not only conceived as a middle path between bottom-up and top-down, but also relies on the understanding that categories are not simply organised in hierarchies from the most general to the most specific, but are rather organised cognitively in such a way that categories are located in the middle of the general-to-specific hierarchy. Going up from this level is the generalisation and going down is the specialisation [43, 18].
C5. No life cycle is recommended.
C6. No techniques or methods are recommended. The authors mention the importance of representing the captured knowledge but do not make explicit recommendations as to which knowledge formalism to use. This methodology does not support any particular software as a development tool. The integration with other ontologies is not described, nor is any method recommended to overcome this issue, nor is whether this integration involves extending the generated ontology or merging it with an existing one explained.
C7. The methodology was used to generate the Enterprise ontology [10].
C8. Communities are not involved in this methodology.
C9. For those activities specified within the building stage the authors do not propose any specific method for representing the ontology (e.g. frames, description logic). The authors place special emphasis on knowledge elicitation. However, they are not specific in developing this further.

A.2 The TOVE Methodology

The Toronto Virtual Enterprise (TOVE) methodology involves building a logical model of the knowledge that is to be specified by means of an ontology. The steps involved as well as their corresponding outcomes are illustrated in Fig. 4.6.

C1. The methodology is heavily influenced by the development of knowledge-based systems using first-order logic [36].
C2. No specifics are provided on the activities involved.
C3. The TOVE methodology emphasises competency questions as well as motivating scenarios as important components in their methodology. This methodology is application semidependent as specific terminology is used not only to

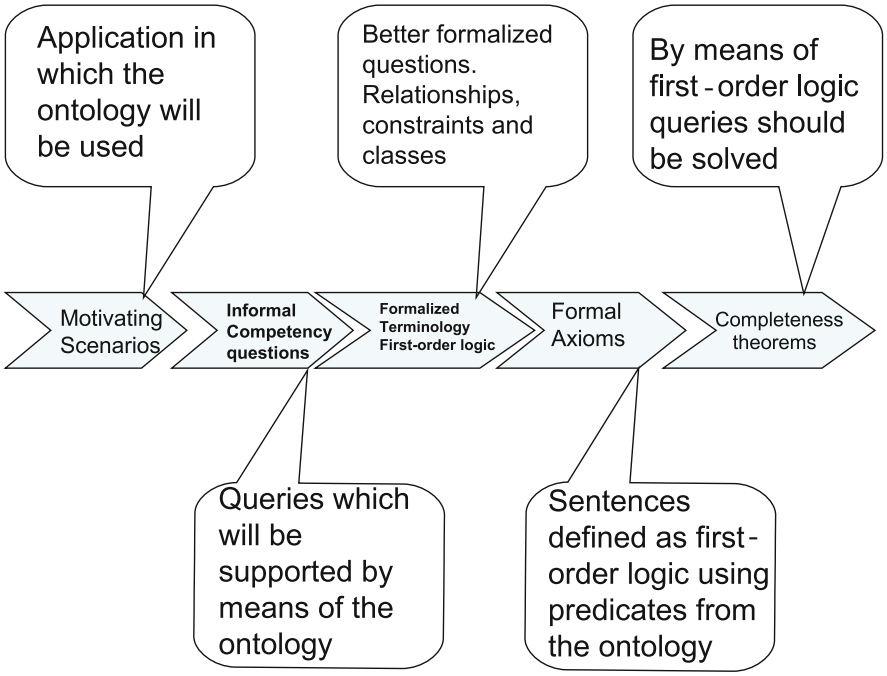


Fig. 4.6 The TOVE methodology

formalise questions but also to build the completeness theorems used to evaluate the ontology. Once the competency questions have been formally stated, the conditions under which the solutions to the questions must be defined should be formalised. The authors do not present information about the kind of domain experts they advise working with.

C4. This methodology adopts a middle-out strategy.
C5. No indication about a life cycle is given.
C6. The importance of competency questions are emphasised. However, they do not provide techniques or methods to approach this problem.
C7. The Toronto Virtual Enterprise ontology was built using this methodology [72].
C8. Communities are not involved in this methodology.
C9. No particular indication for eliciting knowledge is given.

A.3 The Bernaras Methodology

Bernaras work was developed as part of the KACTUS [12] project which aimed to investigate the feasibility of knowledge reuse in technical systems.

C1. This methodology is thus heavily influenced by knowledge engineering.
C2. Limited detail about the methodology is provided.

C3. This methodology is application dependant. As the development of this methodology took place within a larger engineering effort ontologies were being developed hand-in-hand with the corresponding software. This implies that domain experts were being used for both tasks, for requirements interviews and studies as well as for ontology development. This, however, does not mean that domain experts were taking an active role. The authors present very little information about the kind of domain experts they advise working with.
C4. This methodology adopts a bottom-up approach [36].
C5. As the ontology is highly coupled with the software that uses it, the life cycle of the ontology is the same as the software life cycle.
C6. For the specific development of the ontology no particular methods or techniques are provided. However, as this methodology was meant to support the development of an ontology at the same time as the software it is reasonable to assume that some software engineering methods and techniques were also applied to the development of the ontology.
C7. It has been applied within the electrical engineering domain.
C8. Communities are not involved in this methodology
C9. No particular indication for knowledge elicitation is provided.

A.4 The METHONTOLOGY Methodology

The authors of METHONTOLOGY aim to define a standardisation of the ontology life cycle (development) with respect to the requirements of the Software Development Process (IEEE 1074-1995 standard) [18]. The METHONTOLOGY methodology is illustrated in Fig. 4.7.

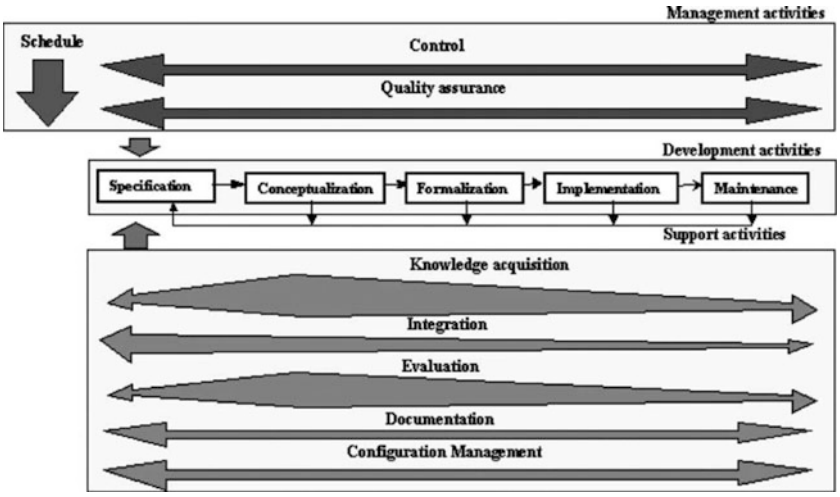


Fig. 4.7 METHONTOLOGY. Reproduced with permission from [36]

C1. METHONTOLOGY has its roots in knowledge engineering.
C2. Detail is provided for the ontology development process; Fig. 4.7 illustrates the methodology. It includes the identification of the ontology development process, a life cycle based on evolving prototypes, and particular techniques to carry out each activity [36]. This methodology heavily relies on the IEEE software development process as described in [50]. Gomez-Perez et al. [52] consider that all the activities carried out in an ontology development process may be classified into one of the following three categories:

    1. Management activities: Including planning, control and quality assurance. Planning activities are those aiming to identify tasks, time and resources.
    2. Development activities: Including the specification of the states, conceptualisation, formalisation, implementation and maintenance. From those activities related to the specification knowledge engineers should understand the context in which the ontology will be used. Conceptualisation activities are mostly those activities in which different models are built. During the formalisation phase the conceptual model is transformed into a semi-computable model. Finally, the ontology is updated and corrected during the maintenance phase [42].
    3. Support activities: These include knowledge elicitation, evaluation, integration, documentation, and configuration management.

C3. Application independent. No indication is provided as to the kind of domain experts they advise working with. In principle METHONTOLOGY could be applied to the development of any kind of ontology.
C4. This methodology adopts a middle-out
C5. METHONTOLOGY adopts an evolving-prototype life cycle.
C6. No methods or techniques recommended. METHONTOLOGY heavily relies on WebODE [45] as the software tool for coding the ontology. However, this methodology is in principle independent from the software tool.
C7. This methodology has been used in the development of the Chemical OntoAgent [73] as well as in the development of the Onto2Agent ontology [73].
C8. No community involvement is considered.
C9. Knowledge elicitation is part of the methodology. However, no indication is provided as to which method to use.

A.5 The SENSUS Methodology

The SENSUS-based methodology [14] is a methodology supported on those experiences gathered from building the SENSUS ontology. SENSUS is an extension and reorganisation of WordNet [74], this 70,000-node terminology taxonomy may[12]

[12] http://www.isi.edu/natural-language/projects/ONTOLOGIES.html

be used as a framework into which additional knowledge can be placed [75]. SENSUS emphasises merging pre-existing ontologies and mining other sources such as dictionaries.

C1. SENSUS is not influenced by knowledge engineering as this methodology mostly relies on methods and techniques from text mining.
C2. Although there is extensive documentation for those text-mining techniques and developing structures for conceptual machine translation [75–77] no detail is provided as for how to build the ontology.
C3. As SENSUS makes extensive use of both text mining and conceptual machine translation the methodology as such is application semi-independent. The methods and techniques proposed by SENSUS may, in principle, be applied to several domains.
C4. SENSUS follows a bottom-up approach. Initially instances are gathered, as the process moves forward abstractions are then identified.
C5. No life cycle is identified; from those reported experiences the ontology is deployed on a one-off basis.
C6. Methods and techniques are identified for gathering instances. However, no further detail is provided.
C7. SENSUS was the methodology followed for the development of knowledge-based applications for the air campaign planning ontology [78].
C8. No community involvement is considered.
C9. Knowledge elicitation is not considered explicitly.

A.6 DILIGENT

Diligent (DIstributed, Loosely controlled and evolvInG Engineering of oNTologies) was conceived as a methodology for developing ontologies on a community basis. Although the DILIGENT approach assumes the active engagement of the community of practice throughout the entire process, it does not give extensive details. Some particularities may be found reported for those cases in which DILIGENT has been used, for instance [15].

C1. DILIGENT is influenced by knowledge engineering as this methodology has been developed assuming the ontologies will be used by knowledge-based systems. However, DILIGENT introduces novel concepts such as the importance of the evolution of the ontology and the participation of communities within the development and life cycle of the ontology.
C2. DILIGENT provides some details specifically for those developments in which it has been used.
C3. DILIGENT is application dependant. There is no indication about the kind of domain experts they advise working with.
C4. The selection between top-down, bottom-up or middle-out is problem dependent. No indication is given as to which strategy would be best to follow.

C5. DILIGENT assumes an iterative life cycle in which the ontology is in constant evolution.
C6. In principle DILIGENT does not recommend methods or techniques. By the same token DILIGENT is not linked to any software supporting, either the development or the collaboration.
C7. Some cases for which DILIGENT has been used have been reported, for instance, the study of legal cases [79].
C8. The involvement of communities is considered in this methodology.
C9. Although knowledge elicitation is considered in this methodology no special emphasis is placed on it.

A.7 The GM Methodology

The GM methodology emphasises on knowledge acquisition when developing ontologies within decentralised settings. Similar to DILIGENT, the GM methodology was engineered for scenarios in which geographically distributed domain experts were working together on the same ontology. The GM methodology makes use of conceptual maps to support the acquisition of knowledge. In contrast to the DILIGENT methodology, the GM methodology provides a detailed description of the process applied within their development scenario.

C1. GM applies knowledge engineering principles.
C2. A detailed description of the methods and techniques used are provided.
C3. GM is application dependant. GM assumes the participation of both specialised and broader knowledge domain experts.
C4. A top-down approach is applied within GM.
C5. GM assumes an iterative life cycle in which the ontology is in constant evolution.
C6. Methods and techniques for some of the stages of the development process are recommended.
C7. GM has been used within the biomedical domain [79].
C8. GM assumes an active participation of the community.
C9. GM has an emphasises on knowledge elicitation.

A.8 The iCapturer Methodology

The GM methodology emphasises on knowledge acquisition within decentralised settings. Unlike GM and DILIGENT, iCapturer [38] makes use of text-mining approaches, such as text-to-onto, to identify important terms and to suggest candidate ontological relationships between them.

C1. The iCAPTURer approach has received little influence from knowledge engineering.
C2. The iCAPTURer methodology is very specific in terms of the orchestration of methods used. The first step is term and relationship extraction from text containing domain knowledge. The second is web-based, massively collaborative correction, refinement and extension of the automatically extracted concepts and relationships. The second step may be divided into phases of knowledge elicitation, evaluation, and aggregation.

C3.1. *Application of the ontology*. Application independent.
C3.2. *Domain experts*. The methodology is intended to make use of knowledge gathered from all levels of domain experts. It is assumed that the pool of experts contains all of the knowledge that is intended to be represented in the ontology.
C3.3. *Ontology type*. The methodology is best suited for domain ontologies.

C4. *Strategy for identifying concepts*. The strategy for identifying concepts is to extract representative terms automatically from text. Though this will typically result in what appears to be a more bottom-up approach, different bodies of text will produce different results.
C5. *Recommended life cycle*. The recommended life cycle is to

1. identify a domain of knowledge to be represented in an ontology,
2. identify a corpus thought to contain that knowledge,
3. apply text-mining approaches, such as text-to-onto, to identify important terms and to suggest candidate ontological relationships between them,
4. define user-interfaces for correcting and extending this knowledge,
5. assemble a broad array of experts in the domain and engage them in using the interface to improve the ontology,
6. evaluate the quality of each contributor based on expected correct interactions with the knowledge elicitation system,
7. weight their contributions based on this level of quality,
8. aggregate the contributions of all of the experts so that a candidate ontology can be generated and
9. iterate and refine as needed.

C6. *Recommended methods and techniques*. The methodology specifies the process but does not suggest any specific method. Several text-mining algorithms or knowledge gardening interfaces might be applied depending on the domain and the community.
C7. *Applicability*. iCAPTURer has not yet been applied in real scenarios.
C8. *Community involvement*. The community is assumed to develop the ontology.
C9. *Knowledge elicitation*. Knowledge elicitation is an integral part of the methodology. iCAPTURer describes some techniques for KE, but there is also wide room for expansion and adaptation of other methods.

A.9 NeOn Methodology

NeOn[13] is a framework for developing networked ontologies. It is one of the most comprehensive works in terms of ontology engineering. The framework incorporates a methodology.

C1. Highly influenced by knowledge engineering.
C2. It defines those steps that should be undertaken when developing ontologies.
C3.

- C3.1. *Application of the ontology*. Application independent.
- C3.2. *Domain experts*. It assumes an active participation of domain experts and ontology engineers.
- C3.3. *Ontology type*. The methodology is best suited for domain ontologies.

C4. *Strategy for identifying concepts*. No particular detail is provided for identifying concepts.
C5. *Recommended life cycle*. Project aims specifically to support life cycle activities, but does not prescribe a particular type of life cycle.
C6. *Recommended methods and techniques*. It provides specifics for methods and techniques.
C7. *Applicability*. The methodology is proposed based on cases that have been studied; however, it is not clear which ontology has been developed applying the proposed framework.
C8. *Community involvement*. It assumes collaboration and the involvement of a community of practice.
C9. *Knowledge elicitation*. Knowledge elicitation is recognised to play a significant role during the development process.

References

1. Editorial: Compete, collaborate, compel. Nat Genet 39(8) (Aug 2007) 931
2. Julian, S., J. Rector, A.: The state of multi-user ontology engineering. In: Proceedings of the 2nd International Workshop on Modular Ontologies, Canada (2007)
3. Smith, B., Ashburner, M., Rosse, C., Bard, J., Bug, W., Ceusters, W., Goldberg, L., Eilbeck, K., Lewis, S.: The obo foundry: Coordinated evolution of ontologies to support biomedical data integration. Nature Biotechnology 25(11) (2007) 1251–1255
4. Gruber, T.: Collective knowledge systems: Where social web meets the semantic web. In: Proceedings of the 5th International Semantic Web Conference, Athens, GA, USA (2006)
5. Tudorache, T., Noy, N.: Collaborative protege. In: Social and Collaborative Construction of Structured Knowledge, Proceedings of 16th International WWW Conference, Alberta, Canada (2007)
6. Feigenbaum, E., McCorduck, P.: The Fifth Generation. Addison-Wesley, Reading, MA (1983)

[13] http://www.neon-project.org

7. Kendal, S., Creen, M.: An Introduction to Knowledge Engineering. Springer, New York, NY (2007)
8. Sowa, J.: Knowledge Representation: Logical, Philosophical, and Computational Foundation. Brooks Cole Publishing, Pacific Grove, CA (2000)
9. Sure, Y.: Methodology, Tools and Case Studies for Ontology Based Knowledge Management. PhD Thesis, Universitat Fridericiana zu Karlsruhe (2003)
10. Uschold, M., King, M.: Towards methodology for building ontologies. In: Workshop on Basic Ontological Issues in Knowledge Sharing, Held in Conjunction with IJCAI-95. Cambridge, UK (1995)
11. Gruninger, M., Fox, M.S.: The role of competency questions in enterprise engineering. In: Proceedings of the IFIP WG5.7 Workshop on Benchmarking – Theory and Practice, Trondheim, Norway (1994)
12. Bernaras, A., Laresgoiti, I., Corera, J.: Building and reusing ontologies for electrical network applications, 12th European Conference on Artificial Intelligence ECAI. Wiley, Budapest, Hungary (1996) 298–302
13. Fernadez-Lopez, M., Perez, A.G., Pazos, S.J., Pazos, S.A.: Building a chemical ontology using methontology and the ontology design environment. IEEE Intelligent Systems and Their Applications 14 (1999) 37–46
14. Swartout, B., Ramesh, P., Knight, K., Russ, T.: Toward distributed use of largescale ontologies. In: Symposium on Ontological Engineering of AAAI, Stanford, California (1997)
15. Pinto, H.S., Staab, S., Tempich, C.: Diligent: Towards a fine-grained methodology for distributed, loosely-controlled and evolving engineering of ontologies. In: European Conference on Artificial Intelligence, Valencia, Spain (2004) 393–397
16. Vrandecic, D., Pinto, H.S., Sure, Y., Tempich, C.: The diligent knowledge processes. Journal of Knowledge Management 9(5) (2005) 85–96
17. Garcia, C.A., Rocca-Serra, P., Stevens, R., Taylor, C., Nashar, K., Ragan, M.A., Sansone, S.: The use of concept maps during knowledge elicitation in ontology development processes – the nutrigenomics use case. BMC Bioinformatics 7 (2006) 267
18. Mirzaee, V.: An Ontological Approach to Representing Historical Knowledge. MSc Thesis. PhD Thesis, Department of Electrical and Computer Engineering, University of British Columbia (2004)
19. Moreira, D., Musen, M.A.: Obo to owl: A protege owl tab to read/save obo ontologies. Bioinformatics 23(14) (2007) 1868–1870
20. Sathiamurthy, M., Peters, B., Bui, H.H., Sidney, J., Mokili, J., Wilson S.S., Fleri, W., McGuinness, D., Bourne, P., Sette, A.: An ontology for immune epitopes: Application to the design of a broad scope database of immune reactivities. BMC Immunology 1(2) (2005)
21. Bada, M., Stevens, R., Goble, C., Gil, Y., Ashbourner, M., Blake, J., Cherry, J., Harris, M., Lewis, S.: A short study on the success of the geneontology. Journal of Web Semantics 1 (2004) 235–240
22. Berners-Lee, T., Hendler, J., Lassila, O., et al.: The semantic web. Scientific American 284(5) (2001) 28–37
23. Shadbolt, N., Berners-Lee, T., Hall, W.: The semantic web revisited. IEEE Intelligent Systems (2006) 96–101
24. 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 Research (2007)
25. Smith, B., Kumar, A., Bittner, T.: Basic formal ontology for bioinformatics. Retrieved Jul. 12, 2010 from http://www.uni-leipzig.de/~akumar/JAIS.pdf Journal of Information Systems (2005) 1–16
26. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L.: Sweetening Ontologies with Dolce. Lecture Notes in Computer Science (2002) 166–181

27. Herre, H., Heller, B., Burek, P., Hoehndorf, R., Loebe, F., Michalek, H.: General Formal Ontology (GFO) – A Foundational Ontology Integrating Objects and Processes. Onto-Med Report 8
28. Eden, H.A., Hirshfeld, Y.: Principles in formal specification of object oriented design and architecture. In: Proceedings of the 2001 Conference of the Centre for Advanced Studies on Collaborative Research, Toronto, Canada, IBM Press (2001)
29. Pressman, S.R.: Software Engineering, A Practitioners Approach. 5th edn. McGraw-Hill Series in Computer Science. Thomas Casson, New York, NY (2001)
30. Martin, J.: Rapid Application Development. Prentice-Hall, Englewood Cliffs, NJ (1991)
31. Gilb, T.: Evolutionary project management: Multiple performance, quality and cost metrics for early and continuous stakeholder value delivery. In: International Conference on Enterprise Information Systems, Porto, Portugal (2004)
32. Perez, A.G.: Some Ideas and Examples to Evaluate Ontologies. Technical Report, Stanford University (1994a)
33. Gilb, T.: Principles of Software Engineering Management. Addison-Wesley Longman, Boston, MA (1988)
34. Garcia, A.: Developing Ontologies Within the Biomedical Domain. PhD, University of Queensland (2007)
35. Fernandez, M.: Overview of methodologies for building ontologies. In: Proceedings of the IJCAI-99 Workshop on Ontologies and Problem-Solving Methods(KRR5), Stockholm, Sweden (1999)
36. Corcho, O., Fernadez-Lopez, M., Gomez-Perez, A.: Methodologies, tools, and languages for building ontologies. Where is their meeting point? Data and Knowledge Engineering 46(1) (2003) 41–64
37. Fernandez, M., Gomez-Perez, A., Juristo, N.: Methontology: From ontological art to ontological engineering. In: Workshop on Ontological Engineering. Spring Symposium Series. AAAI97, Stanford (1997)
38. Good, B., Tranfield, E.M., Tan, P.C., Shehata, M., Singhera, G., Gosselink, J., Okon, E.B., Wilkinson, M.: Fast, cheap, and out of control: A zero curation model for ontology development. In: Pacific Symposium on Biocomputing. Maui, Hawaii, USA. (2006)
39. Van Heijst, G., Van der Spek, R., Kruizinga, E.: Organizing corporate memories. In: Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop (KAW'96). (1996)
40. Mizoguchi, R., Vanwelkenhuysen, J., Ikeda, M.: Task ontology for reuse of problem solving knowledge. In: Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing (KBKS'95). (1995) 46–57
41. Uschold, M., Gruninger, M.: Ontologies: Principles, methods and applications. Knowledge Engineering Review 11 (1996) 93–136
42. Fernadez-Lopez, M., Gomez-Perez, A.: Overview and analysis of methodologies for building ontologies. The Knowledge Engineering Review 17(2) (2002) 129–156
43. Lakoff, G.: Women, Fire, and Dangerous Things: What Categories Reveal About the Mind. Chicago University Press, Chicago (1987)
44. Cooke, N.: Varieties of knowledge elicitation techniques. International Journal of Human-Computer Studies 41 (1994) 801–849
45. Arpirez, J., Corcho, O., Fernadez-Lopez, M., Gomez-Perez, A.: Webode in a nutshell. AI Magazine 24(3) (2003) 37–47
46. Hinchcliffe, D.: Dion hinchcliffe's web 2.0 blog web 2.0 (2008)
47. Stoeckert, C.J., Parkinson, H.: The mged ontology: A framework for describing functional genomics experiments. Comparative and Functional Genomics 4 (2003) 127–132
48. Perez, A.G., Juristo, N., Pazos, J.: Evaluation and assessment of knowledge sharing technology. In Mars, N. (ed.) Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing(KBK95), IOS Press, Amsterdam, The Netherlands, (1995) 289–296
49. Pinto, H.S., Martins, P. J.: Ontologies: How can they be built? Knowledge and Information Systems 6 (2004) 441–463

50. IEEE: IEEE standard for software quality assurance plans (1998)
51. Greenwood, E.: Metodologia de la investigacion social. Paidos, Buenos Aires (1973)
52. Gomez-Perez, A., Fernandez-Lopez, M., Corcho, O.: Ontological Engineering. Springer, London (2004)
53. IEEE: IEEE standard for developing software life cycle processes (1996)
54. Mooney, S.D., Baenziger, P.H.: Extensible open source content management systems and frameworks: A solution for many needs of a bioinformatics group. Brief Bioinform 9(1) (Jan 2008) 69–74
55. Stevens, R., Goble, C., Bechhofer, S.: Ontology-based knowledge representation for bioinformatics. Briefings in Bioinformatics (2000) 398–414
56. Gaines, B.R., Shaw, M.L.Q.: Knowledge acquisition tools based on personal construct psychology. The Knowledge Engineering Review 8(1) (1993) 49–85
57. Rubin, D., Lewis, S., Mungall, C., Misra, S., Westerfield, M., Ashburner, M., Sim, I., Chute, C., Solbrig, H., Storey, M., Smith, B., Day-Richter, J., Noy, N., Musen, M.: National center for biomedical ontology: Advancing biomedicine through structured organization of scientific knowledge. OMICS 10(2) (2006) 85–98
58. Cote, R., Jones, P., Apweiler, R., Hermjakob, H.: The ontology lookup service, a lightweight cross-platform tool for controlled vocabulary queries. BMC Bioinformatics 7(97) (2006)
59. Noy, N.F., McGuinness, D.L.: Ontology Development 101: A Guide to Creating Your First Ontology. Technical Report, Stanford University (2001)
60. Perez, A.G., Fernadez-Lopez, M., Corcho, O.: Ontological Engineering. Computer Sciences. Springer. London (2004)
61. Haarslev, V., Mller, R.: Racer: A core inference engine for the semantic web. In: Proceedings of the 2nd International Workshop on Evaluation of Ontology-based Tools (EON2003), Sanibel Island, Florida, USA (2003) 27–36
62. Sirin, E., Parsia, B., Cuenca-Grau, B., Kalyanpur, A., Katz, Y.: Pellet: A practical owl-dl resoner. Journal of Web Semantics 5(2) (2007)
63. Garcia, A., Zhang, Z., Rajapakse, M., Baker, C., Tang, S.: Capturing and modeling neuro-radiological knowledge on a community basis: The head injury scenario. In: Health and Life Sciences workshop at the WWW2008. (2008)
64. Orchard, S., Hermjakob, H., Apweiler, R.: The proteomics standards initiative. Proteomics 3(7) (2003) 1374–1376
65. Taylor, C., Paton, N., Lilley, K., Binz, P., Julian, R.J., Jones, A., Zhu, W., Apweiler, R., Aebersold, R., Deutsch, E., Dunn, M., Heck, A., Leitner, A., Macht, M., Mann, M., Martens, L., Neubert, T., Patterson, S., Ping, P., Seymour, S., Souda, P., Tsugita, A., Vandekerckhove, J., Vondriska, T., Whitelegge, J., Wilkins, M., Xenarios, I., Yates, J.R., Hermjakob, H.: The minimum information about a proteomics experiment (miape). Nature Biotechnology 25(8) (2007) 887–93
66. Jones, A., Gibson, F.: An update on data standards for gel electrophoresis. Proteomics 7(Suppl 1) (2007) 35–40
67. Dagnino, A.: Coordination of hardware manufacturing and software development lifecycles for integrated systems development. In: IEEE International Conference on Systems, Man, and Cybernetics 3 (2001) 850–1855
68. Boehm, B.: A spiral model of software development and enhancement. ACM SIGSOFT Software Engineering Notes 11(4) (1986) 14–24
69. McDermid, J., Rook, P.: Software development process models. In: Software Engineer's Reference Book. CRC Press, Boca Raton, FL (1993) 15–28
70. Larman, C., Basili, R., V.: Iterative and incremental development: A brief history. Computer, IEEE Computer Society 36 (2003) 47–56
71. May, L, E., Zimmer, A, B.: The evolutionary development model for software. HP Journal (1996) Retrieved Jul. 12, 2010 http://www.hpl.hp.com/hpjournal/96aug/aug96a4.pdf
72. Fox, M.S.: The tove project: A common-sense model of the enterprise systems. In: Industrial and Engineering Applications of Artificial Intelligence and Expert Systems. (1992)

73. Arpirez, J., Corcho, O., Fernadez-Lopez, M., Gomez-Perez, A.: Webode in a nutshell. AI Magazine 24(3) (2003) 37–47
74. Fellbaum, C.: WordNet, An Electronic Lexical Database. The MIT Press, Cambridge, MA (2000)
75. Knight, K., Luk, S.: Building a large-scale knowledge base for machine translation. In: Proceedings of the National Conference on Artificial Intelligence. Wiley, New York (1994) 773–773
76. Knight, K., Chander, I.: Automated postediting of documents. In: Proceedings of the 12th National Conference on Artificial Intelligence (vol. 1) Table of Contents, American Association for Artificial Intelligence Menlo Park, CA, USA (1994) 779–784
77. Knight, K., Graehl, J.: Machine transliteration. Computational Linguistics 24(4) (1998) 599–612
78. Valente, A., Russ, T., MacGregor, R., Swartout, W.: Building and (Re) Using an Ontology of Air Campaign Planning. IEEE Intelligent Systems (1999) 27–36
79. Tempich, C., Pinto, H., Sure, Y., Vrandecic, D., Casellas, N., Casanovas, P.: Evaluating diligent ontology engineering in a legal case study. In: XXII World Congress of Philosophy of Law and Social Philosophy, IVR2005 Granada, May 24th, 29th (2005)
80. Garcia, A.: Developing Ontologies in the Biological Domain. PhD Thesis, University of Queensland (2007)

Chapter 5
Semantic Technologies for Searching in e-Science Grids

Amitava Biswas, Suneil Mohan, and Rabi Mahapatra

Abstract Searching is a key function in scientific cyber-infrastructures; there these systems need to implement superior meaning-based search functionalities powered by suitable semantic technologies. These required semantic technologies should enable computers to comprehend meaning of the objects being searched and user's search intentions, compare these meanings, and discern which object may satisfy user's need. We present a survey of meaning representation and comparison technologies, followed by a design of meaning representation and comparison technique which is coherent to the cognitive science and linguistics models. This proposed design addresses the key requirement of meaning compositionality which has not been addressed adequately and efficiently by existing research. We present an algebraic theory and techniques to represent hierarchically composed concepts as a tensor which is amenable to efficient semantic similarity computation. We delineate a data structure for the semantic descriptors/keys and an algorithm to generate them and describe an algorithm to compute the semantic similarity of two given descriptors (tensors). This meaning comparison technique discerns complex meaning while enabling search query relaxation and search key interchangeability. This is achieved without the need of a meaning knowledgebase (ontology).

5.1 Introduction

5.1.1 e-Science or Scientific Cyber-Infrastructure

Both terms "e-Science" and "scientific cyber-infrastructure" indicate the same concept. The term "cyber-infrastructure" was introduced by National Science Foundation in USA whereas "e-Science" is used in UK. Scientific cyber-infrastructure or e-Science indicates an assortment of digital resources, software

A. Biswas (✉)
Department of Computer Science, Texas A&M University, College Station, TX, USA
e-mail: amitabi@cs.tamu.edu

H. Chen et al. (eds.), *Semantic e-Science*, Annals of Information Systems 11,
DOI 10.1007/978-1-4419-5908-9_5,

systems, and web applications that facilitate and enable activities carried out during scientific investigations. The main argument in favor of such cyber-infrastructures is that if all the data that are being generated in different scientific fields are made available to the scientific community and public at large, then that alone can lead to further scientific discoveries and applications. A hypothetical scenario presented in text box TB1 illustrates this vision.

TB1: Eric, an atmospheric scientist, wants to predict the new climatic patterns in Texas for the next 10 years. He explores the "Texas Mesonet" [1], identifies various required metrological data, receives the data feeds, and inputs them into his model to generate the prediction. He stores his prediction data in his local database server but he allows the grid to produce a basic description about his generated data and computation model and has read access to them.

A few weeks later, an economist discovers and uses Eric's generated data and models along with other available semi-processed data on soil hydrology from CUAHSI Hydrology Information System [2]. He uses these data to estimate the requirements and cost of irrigation for ethanol-grade corn (for wet areas) and sorghum (for dry areas) [9].

Quite a number of such cyber-infrastructures are available today. In bioscience domain NIH's (US) PubMed [1] and EMBL's (Europe) SRS [2] are well-known examples, whereas SAO/NASA Astrophysics Data System [3] is an example in astronomy and Community Data Portal (CDP) [4], California Water CyberInfrastructure [5] and CUAHSI Hydrologic Information System (CUAHSI-HIS) [6] are examples in earth science domains. These cyber-infrastructures enable scientists to (1) acquire new insights and knowledge from existing data; (2) beget collaboration; (3) facilitate reuse of data and knowledge; (4) assist generation of new knowledge from existing ones; and (5) encourage application of the available data and knowledge for productive outcomes. The value of these cyber-infrastructures lies in their usability and functionality. Therefore planning and designing a suitable cyber-infrastructure that can deliver these functionalities to a large scientific user community is a significant challenge.

5.1.2 Scientific Cyber-Infrastructure Planning and Designing Challenges: Scope of Discussion

Cyber-infrastructure planning and design challenges can be categorized into two broad classes: (1) social, behavioral, or "soft" issues and (2) "hard" technical problems that can be addressed by engineering approaches. The planning problems mostly involve social and behavioral aspects of the usage environment. During planning the key question that has to be addressed is how to manage the

usage environment so that the combined emergent system comprising the cyber-infrastructure and its users is most effective in enabling and facilitating scientific investigations. These planning problems encountered here have to be addressed by appropriate changes and innovations in the legal/regulatory framework, organizational designs, etc. Some discussions on these "soft" issues can be found in [7]. Whereas cyber-infrastructure design challenges are mostly technical in nature, the technical side of this design problem involves managing the following three aspects:

1. *Resources*: These include data, information, knowledge in digital forms, computational resources, and computational (software) tools that are shared among users. One of the key roles of the cyber-infrastructure is to enable effective and efficient sharing of these resources.
2. *Processes*: In addition to enabling resource sharing, a cyber-infrastructure should also facilitate other high-level processes that are essential and intrinsic to scientific investigations. These processes comprise a collection of activities and tasks that are necessary to be carried out to explore existing data, information, and knowledge to generate new insights or to re-examine old hypotheses; gather secondary data and/or use computational resources to validate a new hypothesis; etc. The cyber-infrastructure should be designed to have functionalities that enable and facilitate these business processes. This requires business process engineering/designing, which is considered part of information system design activity. Scientific investigation is a creative practice which involves a wide variety of well-established and unprecedented and yet-to-be defined processes. This poses a significant challenge in designing a flexible encompassing system which should preferably enable a large variety of standard and novel unidentified processes.
3. *Technologies*: These include low-level processes, mechanisms, techniques, and algorithms that enable the cyber-infrastructure functionalities. Two broad categories of technologies are involved here, these are core technologies and user interaction technologies. The core technologies are embedded deep within the cyber-infrastructure to support a wide array of functionalities, whereas the user interaction technologies enable easy, user-friendly, and meaningful user interaction between users and the cyber-infrastructure. Both of these technologies are interdependent and together they enable the required processes.

Both aspects of the problem, the planning and technical designing, are also interdependent. Technical design activity should take into account the social (organizational, behavioral), economic, and regulatory realities of the usage context to design effective and efficient processes and technologies. On the other hand, novel cyber-infrastructure technologies enable new functionalities that can facilitate innovative organizational designs and implement new regulations. In this chapter we primarily focus on a specific kind of core semantic technologies that are needed to address some of these cyber-infrastructure design challenges.

5.1.3 Role of Search and Semantic Technologies

Role of search. Search functionality is very central to the "resource reuse" objective of the scientific cyber-infrastructure paradigm. To reuse resources, users have to first discover them (by searching). Search functionality is also essential to support other key functionalities as well. For example, to aid creation of some components of the cyber-infrastructure itself, a superior kind of search capability is required. We discuss these roles of search technology in detail in subsequent sections.

Need for sophisticated meaning-based search capability. The search capability needed here has to be meaning based, more sophisticated than what we get today in Internet search engines like Google and local specialized search engines like PubMed's All search feature [1], etc. Scientists will need to search for data, information, knowledge objects, tools, and computational resources what are more complex and heterogeneous compared to text documents. They will need a more pervasive search service than what is currently provided by the local search engines. It is well known that existing Internet search engines cannot index valuable "deep web" database resources [8] which are more interesting in scientific explorations. In addition, they fail to carry out precise searching to yield fewer and more relevant results even for general search queries (e.g., Table 5.1, where user wanted to identify a supplier of mouse test subjects). Due to these fundamental deficiencies, Internet search engines are not expected to perform well with scientific terms and keywords. On the other hand the specialized search engines do not have the required level of sophistication (an example is presented in Table 5.2).

Table 5.1 Deficiency of Internet search engines (Google, 2008) [9]

Keywords used	Top 20 results	Comments
Rodent supplier	Relevant	Indicates availability of information
Mouse supplier	Irrelevant	Returns computer mouse suppliers (disambiguation problem)
Supplier animal mouse medical experimentation	Irrelevant	Fails to return result even though keyword is disambiguated by adding "animal"

Need for superior semantic technologies. Ability of a search technology to handle large variety of heterogeneous objects, deliver improved search coverage across all Internet sources, and provide precise search results depends on the capabilities of the underlying semantic technology that enables the search. Semantic technologies are "meaning centered" software standards, methodologies, and techniques that enable computers to recognize topics and concepts, extract meaning from data and information, and categorize information and data. The rudimentary semantic technologies of the existing search technologies only allow searches based on broad classification of topics or broad meaning of the object content. To deliver precise search result we need superior semantic technologies that can capture, represent,

Table 5.2 Semantic capability deficiency in NIH's PubMed (as on Jul 29, 2008) [9]

Primary keywords used to derive MeSH [10] terms	Combination of medical subject heading (MeSH) [10] terms used to search in PubMed	Entries returned	Comment
Type 1 diabetes mellitus, PTPN22	"Diabetes mellitus, type 1" [Mesh] and "Protein tyrosine phosphatase, non-receptor type 22" [Mesh]	38	Indicates presence of 38 publications
Type 1 diabetes mellitus, Lyp	"Diabetes mellitus, type 1" [Mesh] and "PTPN22 protein, human " [Substance name]	37	This result should have been the same as above
Type 1 diabetes mellitus, Csk	"Diabetes mellitus, type 1" [Mesh] and "protein tyrosine kinase p50(csk) " [Substance name]	0	Should have given similar results as above rows

"PTPN22" gene encodes Lyp protein, which binds with Csk to inhibit T-cell activation in Type 1 diabetes [75].

and compare meanings more precisely. Semantic web technologies are a sub-class of semantic technologies and the semantic technologies that are required for scientific cyber-infrastructure are distinct from semantic web technologies.

5.1.4 Chapter Overview

In this chapter, cyber-infrastructure (e-Science) being the center of interest, we start by analyzing their generic requirements. We discuss possible roles of search and semantic technologies in satisfying some of these key requirements. Next we review the concepts, performance metrics, user expectations regarding search technology and present a proposed architecture for searching in e-Science grids (scientific cyber-infrastructures). With this background we illustrate that to enable sophisticated search capabilities we need techniques to capture user's search intention, represent meaning of objects, and meaningfully match user's search intention against the meaning of the available objects. We present a brief survey of semantic technologies, cognitive science, and linguistics literature that are relevant for meaning representation and comparison. With this backdrop, we present a proposed set of meaning representation and comparison techniques and design a distributed index mechanism that can enable faster and scalable searching in a large grid. These techniques are orthogonal to semantic web technologies and can be deployed along with them to bolster overall semantic functionalities in scientific cyber-infrastructures.

5.2 Scientific Cyber-Infrastructure: Functional Requirements

The key motivation behind a cyber-infrastructure is that appropriate informatics tools can facilitate and stimulate research. To design appropriate tools it is important to first identify and understand the business processes involved in scientific

investigations. Therefore we identify and discuss these processes as part of the requirement analysis.

5.2.1 Business Processes in Research

Processes in scientific investigations are sometimes indeterminate and often vary widely depending on the domain, nature of problem being investigated, personal style of the investigators, availability of scientific resources (data, tools, and methods), etc. Though low-level processes may vary, the high-level processes are deterministic and generic across multiple domains. Figure 5.1 illustrates typical generic high-level processes in natural science research domain.

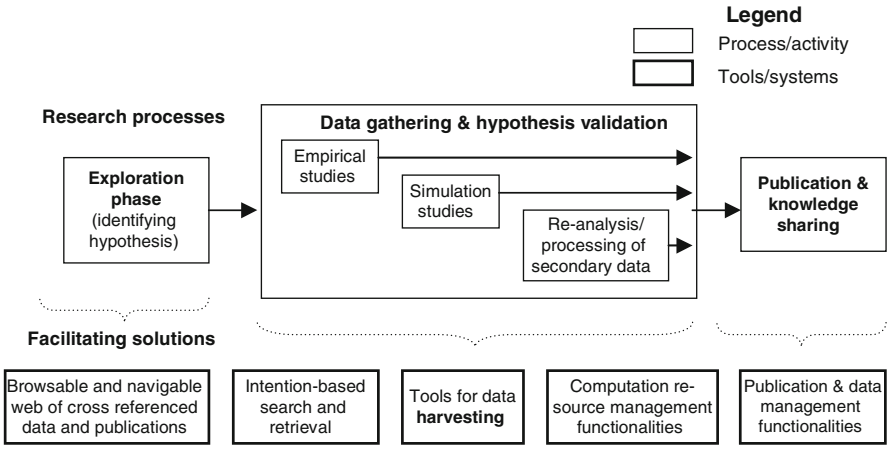


Fig. 5.1 Information technology tools and functionalities to aid research processes [9]

Initially researchers go through an exploratory phase to understand a particular phenomenon and its related aspects to develop an intuition about the phenomenon and to identify a possible hypothesis. Next they try to gather data to prove the hypothesis. There are several alternative and complementary methods. Some of these are (1) carrying out empirical studies under controlled experiments or with real-life uncontrolled observations to gather primary data; (2) undertaking computation-oriented simulations to generate simulated data; (3) gathering and reprocessing existing (secondary) data. With the advent of computers and sophisticated instrumentation there is a greater emphasis on use of computers to produce, capture, analyze, present, and visualize large volume of data to generate insights and knowledge. Therefore capturing/collecting data and preparing data, computational models, and computational tools (software) for computation, marshalling computing resources, etc., have also become part and parcel of the data gathering process.

To share, disseminate, and facilitate reuse of scientific knowledge, scientists publish results, study findings, and insights. These activities constitute the publication

process (Fig. 5.1). With availability of information technology, the publishing is increasingly becoming a web-and-electronic-media-centric process. Therefore these days, the publishing also includes activities like uploading data, computation models, tools, computational objects, etc., to public digital libraries and repositories for sharing with the user community at large. This involves large-scale data curation (preparation, annotation, indexing) and management effort on part of the scientists and repository maintainers.

5.2.2 *Cyber-Infrastructure Functional Blocks and Enabling Technologies*

Each of the identified processes can be facilitated and enabled by appropriate cyber-infrastructure functionalities. These functionalities are identified in Figure 5.1 under the processes that they enable. Figure 5.2 presents a high-level functional architecture comprising functional blocks that embody these functionalities. The

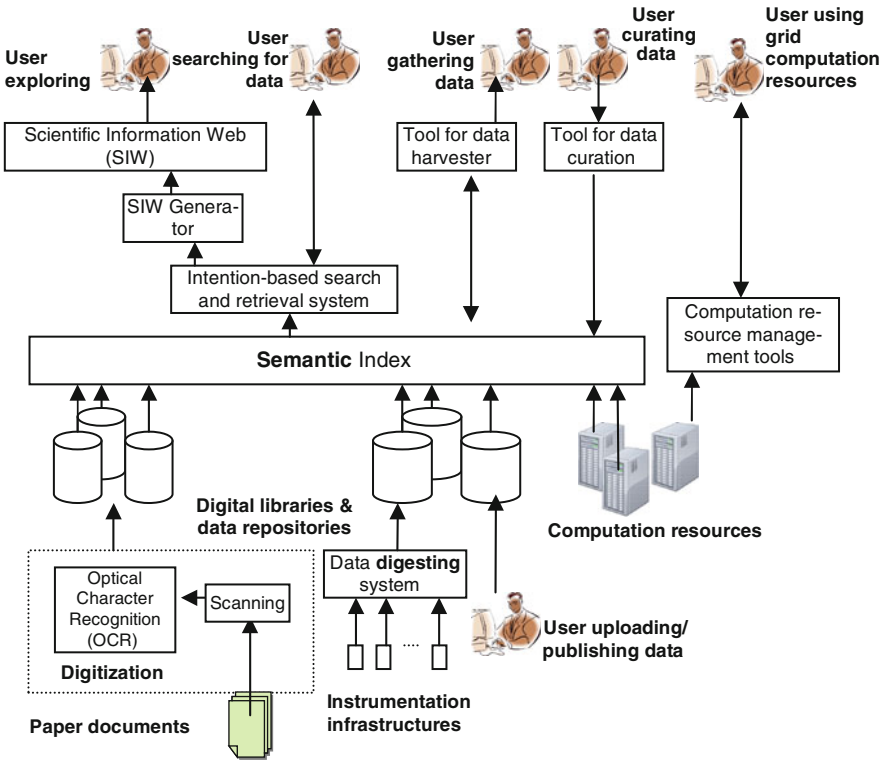


Fig. 5.2 Functional view of a generic scientific cyber-infrastructure [9]

arrows in Fig. 5.2 indicate the flow of information and dependency between functional blocks. These functional blocks and semantic technologies that enable them are discussed below.

Scientific information web (SIW). During exploration phase of the investigation, scientists will want to understand the relationships between various related sub-phenomena and aspects of the problem (Fig. 5.1). This can be facilitated by presenting the data and relevant scientific publications in the form of a navigable and browsable Scientific Information Web-(SIW) where all related resources (data, publications, and tools) are linked by cross-references (Fig. 5.2). The user interface should enable scientists to (1) browse and navigate from one data entity or document portion to another; (2) drill down into each resource object to progressively identify other objects that are related to it; and (3) acquire insights and generate hypothesis about a specific phenomenon. An example of this kind of data presentation is SRS [2].

At present such cyber-infrastructures (cross-referenced web of data and information) are manually constructed. To group related objects, create cross-references and define index terms; data curators and indexers manually search for related objects, index terms and publications (source of knowledge to use), and experts (to get help). This manual and labor-intensive approach to process, curate, and index large volume of data not only suffers from a bottleneck due to paucity of expert data curators but also is prone to errors and omissions. Text box TB2 presents a typical case of such omission.

Sophisticated search technologies can be used to reduce or even obviate some of these manual tasks. Availability of meaningful search capability would allow indexers to quickly identify knowledge sources, experts, and appropriate index terms. It will also help pairing up related objects for cross-referencing (indexing). The omissions in SIW can be avoided by deploying technologies that automate creation of cross-references between related objects. If a sophisticated meaning-based search technology is available it will be possible to automatically group related objects and create cross-references between them (system-assisted cross-referencing and generation of data integration infrastructure) and automatically generate the SIW. The SIW generator will carry out this task.

TB2: When a search related to diabetes biochemistry is carried out in EMBL's SRS [6] with "PPAR" (a nuclear receptor) as the query key, all data objects that are related to "Fmoc-L-leucine" are expected among the results in addition to those related to PPAR, because the "Fmoc-L-leucine" molecule binds to nuclear receptor (protein) named "PPAR" in the insulin signaling pathway (related to diabetes). However, a particular published paper (a data object) related to Fmoc-L-leucine could not be found either by using search keywords "PPAR" or "Fmoc-L-leucine." This paper was only identifiable by the IUPAC name [7] of Fmoc-L-leucine This omission happened because the data curator did not construct the link between "PPAR" and "Fmoc-L-leucine," even though the link to its IUPAC name was put in place [9].

SIW generator. The SIW generator will take the description of a given object and use it as the search key to identify all other objects that are similar (and related) to the given object. This operation will be aided by a sophisticated search and retrieval system and semantic index. The functionalities and components of this search and retrieval system are described below.

Intention-based search and retrieval system. While focusing on a specific aspect of a problem, scientists will prefer to get all related data, information, and knowledge about that topic together at a single place (Fig. 5.1). On the other hand, to allow organic (unfettered) growth and scalability in data collection, management, and storage, data have to be distributed across multiple digital libraries and repositories. To reconcile these two conflicting requirements, a sophisticated intention-based search and retrieval functionality is needed, which can quickly identify all related data from multiple sources and present them to a user on demand. This search system has to be more advanced than existing Internet and conceptual search engines in several aspects. Researchers should be able to describe their search intention and this search system should be able to search multiple public domain repositories to precisely identify the required data in a smaller number of search iterations (Fig. 5.2). The search results can be presented as faceted, clustered, or ranked list [9] based on search results or relationship between them. The closest example to this idea is PubMed's "All search" service. However, this service is not sufficiently sophisticated (due to inconsistencies).

Similar to SRS, PubMed's index is also incomplete as it is manually created and curated. Sometimes a complete set of relevant results are not returned by this service, even when exploded search is carried out using Medical Subject Heading (MeSH) [10]. Table 5.2 presents a typical case of such inadequacy of the conceptual/exploded search capability in PubMed's search service that hinders scientific explorations.

The inconsistencies in Table 5.2 mean that users will miss many relevant data objects unless they have exhaustively tried a large number of search iterations with all possible combinations of all related keywords. There is a risk of missing some keyword combinations and getting an incomplete picture, which is definitely a public health and commercial risk in drug-related research. Sophisticated intention (meaning)-based search capability can address these limitations.

Distributed semantic index. Intention-based searching and sophisticated data presentation will not require building a new repository or duplicating the data storage effort. It will only need a new kind of meaning-based semantic index (catalog) which will refer to data and portions of publications which already exist in public domain digital repositories. This semantic index will be a key component of the cyber-infrastructure grid that will bring together multiple data sources and provide a single-window search service. This index will cluster these objects (e.g., publications or scientific data) in virtual groups based on similarity and relatedness of their content (or its meaning). Based on such a semantic index it will be possible to automatically create cross-references between related objects that are closely grouped together and generate the Scientific Information Web. We need semantic technologies to build this kind of meaning-based index which will drive the intention-based

search functionality and generation of the information web. To achieve scalability and organic growth, this index should be distributed.

Data collection, assimilation, data harvesting tools, and systems. To gather data, scientists will need appropriate software tools to collect, manage, and assimilate (annotate, prepare, and store) data in large scale. They have to gather these data from multiple sources: existing hard copy and electronic publications, data repositories, and instrumentation infrastructures. In some domains, for example in hydrology, a wealth of data and knowledge are available in theses, dissertations, technical reports, conference proceedings, and journal publications. Many of these publications are still in paper or scanned image format. Suitable tools and applications are necessary to seamlessly digitize these paper documents, OCR [11] the scanned images, and meaningfully curate (annotate, index, cross-reference, and cluster) them. This is necessary so that these publications can be searched and data can be harvested from them to create primary and secondary (value-added) data products. Similar tools and systems are also necessary to digest and prepare raw data gathered from small and large-scale experiments, instrumentation, and observatory infrastructures.

Intention-based search technology can enable data indexers and curators to quickly search for background knowledge, relevant search terms and expertise and automate the cross-reference generation tasks. The cross-reference generated automatically can be suggested to experts and they can either approve or reject them. Similarly this search technology can help users to identify relevant portions of the publications from where data can be harvested using suitable tools. When large-scale sophisticated searching is possible then data harvesting from existing publications can become a feasible and economic option in many scientific domains, for example hydrology.

Resource management functionalities. Large-scale simulation and processing often require large-scale computation facilities. Due to economic considerations, generally only a limited number of such facilities are available. This necessitates optimum utilization and sharing of these resources among large number of users. This requires appropriate administration, resource allocation, and management functionalities in the cyber-infrastructure that avoids elaborate bureaucratic administration mechanisms. Searching for available computing resources is also a search problem, where need for meaning-based searching is not as critical. Nevertheless, with availability of the distributed semantic index even this problem can be easily addressed.

Publication support tools to ease data sharing. Scientists generate derived data from the gathered data for their research. Both the primary and derived (secondary) data can be repurposed if they are available to the scientific community and public at large. Although many of these scientific data have resulted from publicly funded projects, lack of incentives, existing reward systems, lack of publication facilitating tools and mechanisms kept researchers from suitably organizing and publishing their data. Various projects have demonstrated that availability of data publication tools and mechanisms can change this culture and encourage scientists to publish their data. This means that cyber-infrastructures have to incorporate the functionalities that encourage scientists to publish their data in addition to traditional scientific publications.

Publication in electronic form means making the published object available in a form that is easily searchable. The crucial task in the publication process is creation of search handles (index keys) and inclusion of these along with the object locations in the index. Existing search technologies require that data have to be elaborately and manually prepared and annotated; metadata should be created with controlled index terms and linked with existing objects in the libraries. Tools which automate these tasks are needed here.

We discussed earlier how search technologies can help partial automation of some of these data curation tasks. In addition, another approach is possible which can automate the indexing task to a greater extent. In this scheme the publisher of the object has to upload the data, preferably in standard format, to a web server which need not be a specialized digital library. In addition, the publisher has to also upload a well-articulated description of the object in natural language (e.g., English). Web crawlers and indexers powered by suitable semantic technologies will identify suitable index terms and generate a semantic descriptor that represents the precise meaning of the object and include that descriptor in the distributed semantic index. The index will use these semantic descriptors for its operations. This operation requires a suitable semantic descriptor generation technology which has to be more sophisticated than the vector-based approaches (e.g., TF–IDF [12]) used today by Internet search engine web crawlers. Researchers can provide further specific annotations and index terms if they wish to do so using the data curation tools.

Integrated operations under cyber infrastructure. All these identified tools and systems will have to be integrated under a single cyber-infrastructure as presented in Fig. 5.2. Researchers will explore, search, and gather data from the cyber-infrastructure. They will use the cyber-infrastructure to marshal computing resources to run their data processing and computation jobs. Finally they will publish their data using the cyber-infrastructure tools. This infrastructure will continuously enhance its capabilities as more users use and contribute to it. These searching, curation, data harvesting, and publishing tools will facilitate appropriate storage and use of data, information, and knowledge. This will lead to further discoveries, generations of new knowledge, and high-impact applications.

5.3 Search Technology for Cyber-Infrastructures

In the last section we discussed how search technology can play a key role in cyber-infrastructures. To understand the requirements for the search technology, in this section we review the basics, evaluation criteria, and user expectations regarding search.

5.3.1 Search Basics

Collection of objects and keys. A database being searched is considered as a collection of key and object/record pairs (Fig. 5.3). The object is the data item or entity which we want to retrieve. In database systems, a record is an object, whereas in

the web an object is less structured and more heterogeneous. The object may be a document file, entire or part of a webpage, a Web site, or web service URL. The key is the identifier or handle for an object which is used in the search process. This key may be a single numerical value, text string as in case of relational database systems, a set of descriptive keywords to tag pictures or video files, or a vector of keywords/terms as used in vector space information retrieval models. In essence, this key is a descriptor that is a compact and structured description of the object which enables retrieval.

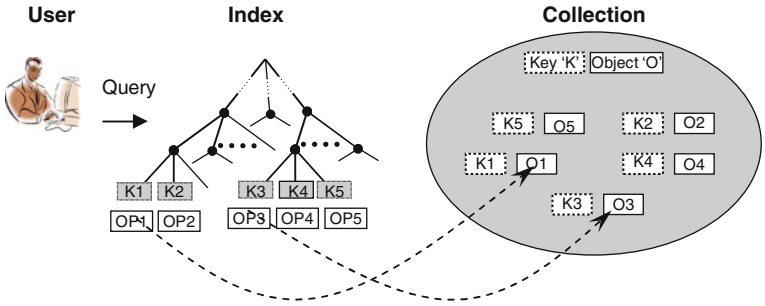


Fig. 5.3 The existing search paradigm [9]

Search query and key comparison. To search, the user has to provide a search criterion. This search criterion is represented by an entity (expression) called search query. The search engine effectively compares all keys to identify which objects have keys that satisfy this criterion and presents those objects as results. Comparing numerical or text string keys is straightforward, but comparing complex descriptors requires sophisticated algorithms. Numerical value and text string key comparisons result in discrete Boolean values 0 (not equal) or 1 (equal), but complex descriptor comparisons generate continuous values. In those cases a higher comparison value means greater similarity between the object and the search key.

Role of Index. To enable faster searching across larger collections, pre-computed indexes are used. Index serves the function of a catalog in a library. It is a data structure of key and object pointer pairs, which is ordered based on the key. The ordering and organization of the index drastically reduces the number of comparisons required to search the entire collection. For example, the hierarchical index structure in Fig. 5.3 enables searching using only ($\log N/\log k$) comparisons instead of N comparisons (in case no index is used), where $N =$ number of objects and $k =$ the number of children in each index tree node.

Result presentation. Database searching produces results that exactly satisfied the user's search criterion, whereas the results from the advanced search technologies are a collection of probable items that finally may or may not match user's need. The best way to reduce user's effort is to rank the results in terms of relevance so that users can examine only the top few results to satisfy their need. This requires computing the relevance of searched items and rank the items. In some

cases, for example, in vector space models, the relevance computation is based on descriptor comparison that yields continuous values. Objects whose descriptors are more similar to the search key are attributed with higher relevance values. In these cases, the key similarity computation itself gives the relevance metric to rank results [9].

Alternate ways to present results are also possible. When users cannot produce a precise search criterion, then large amounts of results can be returned by the search systems. To help users to explore, discover, and refine their search criterion this large volume of results can be presented as clusters (e.g., www.kartoo.com) and facet-based categorizations (e.g., www.illumin8.com). The semantic technologies that power the search technologies are expected to support some of these alternative forms of result presentations expected by users.

5.3.2 Search and Retrieval Performance Metrics

The following metrics are useful to evaluate the performance of any search and retrieval systems.

Precision. It is the fraction of the retrieved objects that are relevant to the user. This metric is calculated as

$$\text{precision} = \frac{\text{number\_of\_retrieved\_objects\_that\_are\_relevant}}{\text{number\_of\_all\_retrieved\_objects}}$$

Recall. It is the fraction of the available relevant objects in the collection that were successfully retrieved. This metric is computed as

$$\text{recall} = \frac{\text{number\_of\_retrieved\_objects\_that\_are\_relevant}}{\text{number\_of\_all\_relevant\_objects\_in\_the\_collection}}$$

Search response time. This is the time a user has to wait to get all objects from the retrieval system to satisfy his/her need. The user does not have to wait for all available objects to be retrieved; he/she has to wait for a shorter duration to get the minimal amount of objects that will just satisfy his/her needs.

Complete recall (response) time. This is the time required to complete the search and recall all the available and relevant objects in the collection. This duration is larger than search response time. The two response time metrics are not traditionally used as retrieval performance evaluation criteria. So far information retrieval has been limited within relatively small centralized systems, where these two times are small enough to be ignored. However, these two time responses will become significantly large and distinct in large distributed search systems which are needed for grid searching. Hence we require to measure and manage both of them through superior grid search system designs.

5.3.3 Existing Meaning-Based Search Techniques

For searching heterogeneous objects in grid, meaning-based searching is needed. There are several varieties of meaning-based searching available. Some are described below.

Conceptual (exploded) searching. This kind of searching expands the scope of search by automatically including search keywords that are conceptually related to the given search/query keywords/terms. An example is PubMed's exploded search service [13] where a user can provide a standard Medical Subject Heading [10] "Glucose Metabolism Disorder" as the search key to get all bio/medical science publications (objects) indexed under that topic. The results will also include objects that are indexed under topics like "Diabetes Mellitus," "Glycosuria," "Hyperglycemia," in addition to those that are indexed under "Glucose Metabolism Disorder." In medical science the concept "Glucose Metabolism Disorder" encompasses "Diabetes Mellitus" or "Glycosuria"; therefore the search is expanded to include all hyponyms of "Glucose Metabolism Disorder." The hyponym–hypernym relationships between index terms are based on controlled taxonomies.

Semantic searching. This kind of searching has the flexibility to tolerate interchangeable search terms as long as these terms broadly convey similar meanings (semantically related). This is best explained with an example as follows.

The gene "PTPN22" has several (single nucleotide polymorphic) variants: 1868C, 1858T, etc. The gene "PTPN22" is also known by synonyms "IPI00298016," "PTPN8: Tyrosine-protein phosphatase non-receptor type 22." PTPN22 is related to a protein "LyP," whose function is to bind with another molecule known as "CsK." Therefore all these terms "PTPN22," "CsK," "LyP," and "1858C," are related. When these related terms are used as interchangeable search keywords the semantic search engine should retrieve a similar set of objects as results in all cases. The semantic relationships used here are available in upper or domain-specific lower ontologies or word association (semantic) networks [14] that can be generated by mining from a corpus of texts on the related subject matter.

5.3.4 Intention-Based Web Searching

Basic concept. Intention-based searching is more sophisticated than the existing meaning-based searching. In this paradigm a user need not provide the exact description (matching keywords) of the data objects or its schema, instead he/she can provide a natural language description of the search intention to the user interface. The search system will search all public domain sources to precisely identify the required information. It will produce a few useful accurate results instead of a long list of probable results. Natural language is the easiest way for a user to describe his search intentions accurately, hence it is a preferable method for interaction with the search system. However, alternate user interaction and result presentation interfaces like result clustering, faceted categorization, cross-referenced web of data are

also possible [9]. Intention-based search will involve capturing meaning of user's intentions and the text description of the objects (or the text objects themselves) and then comparing these meanings to identify which texts/objects will suit user's need.

Implementation approach. To implement intention-based web searching we need to extend the notion of key, key comparison, and index. Intention-based searching involves comparing the meaning of user's intention against description of the object. Therefore the key should be a semantic descriptor data structure that represents the meaning of user's intention and the object. Key comparison should ascertain the similarity between meanings represented by two semantic descriptors. Here the semantic descriptor (key) will replace natural language in representing and comparing the meaning within computers. The index should enable faster searching. In addition the index should be an infrastructure that can improve the search coverage. Construction of distributed meaning-based index is possible by adopting the techniques illustrated in [15, 17] which are based upon the findings of [16]. However that will require techniques to represent and compare meanings of objects within computers.

5.4 Semantic Technologies: Requirements and Literature Survey

5.4.1 Crucial Semantic Technologies

To enable the construction and operation of a meaning-based index system the following semantic technologies are necessary:

1. An advanced data structure to represent keys (descriptors) that represent meanings.
2. Technique to generate this semantic descriptor data structure from natural language texts.
3. Method to compare two descriptor data structure for similarity in their meanings.

The design requirements and the designs for these artifacts are explained in the next section.

5.4.2 Requirements for Semantic Technologies

Design of descriptor data structure and techniques to generate descriptors and methods to compare them should address the following requirements:

1. *Descriptor data structure should be able to express complex concepts (or meanings).* Human beings convey (narrate) and comprehend meanings using concepts. Concept is the mental representation of meaning in human mind, so comparison

of meaning actually means comparison of concepts. Descriptions of objects can be quite complex involving complex concepts. For example, the phrase "color of the leaf is green" conveys a complex concept which human beings comprehend and visualize in terms of elementary concepts such as the physical object "leaf," its "color," and the instance of color "green." Each elementary concept itself can be a hierarchical composition of more elementary concepts under it. Therefore descriptor data structure should enable hierarchical concept composition. The elementary concepts may not always be available and mentioned in the text and may have to be derived from tacit knowledge depending on the context.

This is a key requirement because this compositionality aspect is fundamental in meaning processing and meaning comprehension. The evidence of this compositionality aspect of human thought is available from behavioral, cognitive sciences, and neurological research findings. Behavioral studies indicated that humans genuinely think in terms of combinatorial thought structures [18]. Humans combine elementary thoughts and ideas to generate more complex ones (composition of concepts), which form the basis of reasoning and learning. Neuro-scientific [19–22] and linguistic [23, 24] evidences of semantic (meaning) composition are also available.

2. *The descriptor design should be coherent with human cognitive processes* and supported by cognitive science understandings. This requirement is needed to ensure realistic data structures to serve the needs of practical applications.
3. *The meaning represented by the text should be seamlessly transferable to the descriptor without loss of any information.* This is necessary for easy generation of unambiguous and accurate descriptors from text.
4. *The descriptor should unambiguously capture the entire meaning represented by a narrative and the similarity comparison algorithm should compare meanings of the narratives instead of only sentence or terms.* This is needed to ensure that different texts having similar narrative meanings but having different sentences and syntactic compositions should generate similar descriptors.
5. *The semantic similarity comparison should be a self-contained process* and the descriptor should be sufficiently descriptive to preclude the need of additional information or knowledge to disambiguate meaning.
6. *Descriptor data structure should be compact for efficient storage in the index and transmission as message payloads*.
7. *The similarity computation technique should be computationally efficient* for faster comparison during index operations (lookups, additions/deletions, etc.).

5.4.3 A Study of Meaning in Human Cognition and Language

To enable intention-based searching, we need a mechanism to capture, represent, and compare meanings within computers. Therefore it is important to understand how human beings comprehend meaning and use languages to convey meanings. This will need integrating, developing, and applying notions assimilated from cognitive science, neuroscience, linguistics, anthropology, and mathematics. In this

section we discuss the ideas and evidences that are the key premises for the design of a descriptor data structure that can represent and enable comparison meaning.

Meaning comprehension is distinct from language syntax processing. Here we present evidence and argue that the thought process which involves comprehending complex concept (semantic processing) is distinct from interpreting a sentence using its part of speech components (syntactic processing). This is supported by the fact that non-linguistic species like primates, apes, and human children who do not have language ability have complex thoughts, reasoning, and learning capabilities [18].

Several other evidences also suggest this separation between syntactic and semantic abilities of human brain. Broca's aphasia is caused by damage to some particular areas of the brain known as Broca's area [21, 25–27]. These aphasia patients can comprehend complex meaning which indicates presence of complex thoughts. But they have severe difficulty in communicating them using language or in interpreting language utterances or writings [27]. This situation is different from Wernicke's aphasia, which is caused by damage of Wernicke's area of the brain which is distinct from Broca's area [21, 25]. Wernicke's aphasia patients speak fluently but their words do not have any meaning and the patients have difficulty in comprehending and discerning meaning from sentences.

Brain scans and other neurological studies also indicated that interpretation of language (syntactic processing) and meaning comprehension (semantic processing) are two distinct neurological processes that use different parts of the brain and neural pathways [19, 20, 22, 28–30]. However during language processing these two neurological processes challenge and test each other for incongruence or look for support and cues in case of ambiguity [22, 28]. Studies on fluent bilinguals, monolinguals, monolinguals with different levels of second language competence suggest that though sites for syntax processing vars for different languages and competency levels, the neural site, which is used for semantic processing, is common. Interestingly this common site is also the Wernicke's area of the brain [31, 32]. All these evidences indicate the importance of meaning composition in meaning (semantic) processing and comprehension within human mind.

Principles of meaning composition. All the above–mentioned evidences are also coherent with the "simpler syntax hypothesis" [24, 33] and "parallel architecture" theory [18] from linguistics. These hypotheses argue that in human mind, semantics or meaning has its own rules of composition, which are different from grammatical composition (syntactic) rules. This is based on the observation that sometimes language syntax alone does not sufficiently represent the entire meaning, but yet humans understand the full meaning. For example, the sentence

"*Jane attempted to pass the test*" [9]

invokes more than one meaning. There is a tacit meaning which indicates that Jane took a test, in addition to the explicit one which is about her attempt to pass. The compositional principles that govern composition of semantics are not always evident in the language's explicit syntactic representation, but nonetheless they definitely come into play during communication, just before language generation,

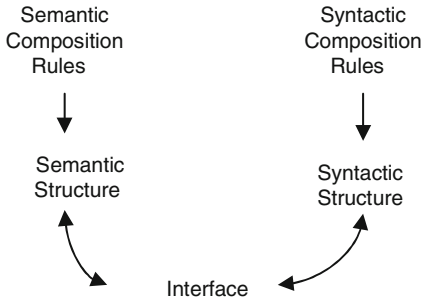


Fig. 5.4 Semantic and syntactic processing is distinct [9]

or during language comprehension. So there must be another parallel composition process (Fig. 5.4) which is taking place in the human mind in addition to syntactic processing. This notion is fundamental to language generation and meaning comprehension [34].

Meaning composition as a simple collection of elementary concepts. Simpler syntax hypothesis [18, 24] proposed that rules for composing meaning are inherently simpler. A simple collection of elementary meanings is good enough to represent a complex meaning. There is no need to specify the ordering of the elements or the exact nature of association between individual elements. This is supported by evidences like presence of compound words in modern natural languages and speech of children, pidgin language speakers, and late language learners who communicate complex notions with a collection of simple words and terms which are devoid of any grammatical relationship. Some examples are illustrated below:

Children: "*Walk street; Go store*", "*Big train; Red book*" [35]
Late language learner raised in the wild: "*Want milk", "Big elephant, long trunk*" [35]
Pidgin language: "*And too much children, small children, house money pay*", "*What say? Me no understand*" [35]
English compound words: "*winter weather skin troubles*", "*health management cost containment services*", "*campaign finance indictment*" [36]

All of these utterances are just collections of words each of which conveys simple elementary concept. These collections of words together convey a complex concept. Here the meaning composition is happening without aid of sophisticated syntactic (grammatical) rules. This indicates that a simple collection of words by itself can represent a complex meaning. This notion is also directly supported by [30]. Here the words stimulate elementary thoughts, meanings, or concepts in human mind which then combine and invoke the complex meaning [23]. References [28, 35] proposes that ability for this kind of semantic composition is intrinsic to human brain and this is also the basic tier of language comprehension ability.

Generative mechanism for semantic composition: The parallel architecture notion emphasizes that a generative mechanism for semantics also exists [18]. This generative mechanism allows application of a simple set of composition rules to

generate very complex structures from simpler elementary structures. There are two simple rules, one that allows representation of complex meaning as a collection of elementary meanings and the other that allows composition of a collection of meanings to form higher level collections. Using these two rules it is possible to represent a complex meaning as a hierarchical collection of elementary meanings (Fig. 5.5). This representation of complex concept (meaning) is a tree structure which can represent complex meaning (concept) in terms of elementary concepts at the leaves (Fig. 5.5) [9].

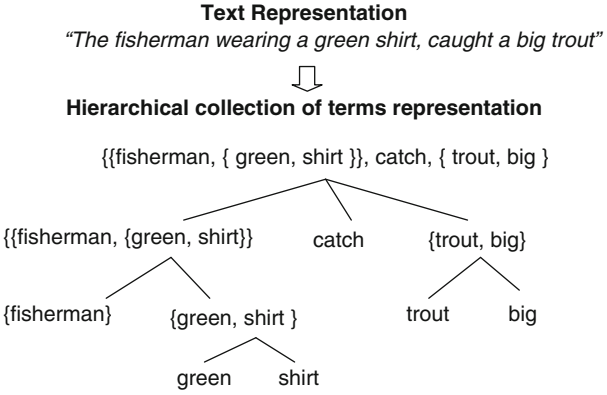


Fig. 5.5 Hierarchical collection of words representing complex meaning [9]

Simple composition and memory models in cognitive science. Collection of concepts as a suitable representation of complex concepts is also supported by cognition science research like [30]. This model is coherent with the spread activation model of associative semantic memory [37–39]. Each of the elementary thoughts stimulates independent spread activations in the human semantic memory in which all of such activations acting together finally give rise to an thought in an associative semantic memory [37]. This spread activation model explains how a collection of elementary concepts can invoke a complex meaning.

This also corroborates well with the semantic memory-related empirical observations made by [40, 41]. These studies indicate how multiple elementary concepts are useful in retrieving complex concepts through node activations and how these complex concepts are manipulated in human mind using these elementary concepts as search handles. In fact a larger number of elementary concepts are likely to generate better activation (recalling) or invocation of a complex concept.

Need for a mathematical model to represent and compare meaning. A basic hierarchical tree structure is reasonably good to represent a complex meaning. To use this model to compare meaning inside computers we need mathematical logic and a computational model to compare two such trees. In the following section we present some previous research that has been done in the past to represent meaning within computers.

5.4.4 Meaning Representation in Computers: Existing Works

Only a few designs for semantic descriptor data structures are available in published literature. Descriptors for objects and also concepts are both necessary and most of the designs are either descriptors for objects or for concepts. The object descriptor designs that are available in literature are based on vector, set, and Galois lattice [42, 43]-based data structures, whereas the concept descriptor design reported in [44] is based on graphs. On the other hand Biswas et al. [45] present a design that unifies the notion of object and concept descriptors. This particular design views the entire object description as a large complex concept. Therefore the object descriptor can be represented by a concept descriptor. We present this idea later.

Set-and vector-based descriptors are based on elementary terms contained in the (natural language) text description of the object. From this text, important information-bearing terms are chosen to construct a vector or set which serves as the object's descriptor. To enable semantic search using vector-and set-based descriptors, it is necessary to have techniques to compare the semantic similarity between elementary terms. A large number of term-level semantic comparison techniques are available in the literature, but vector-and set-based designs are the most adopted ones. We explain the fundamental notions behind these designs and the term-level semantic comparison techniques in detail because they are also behind a more advanced descriptor design presented later. All these designs and their criticisms are discussed in the subsequent paragraphs [9].

Galois lattice-based object descriptor. This design is based on Formal Concept Analysis (FCA) theory [46] from mathematics and statistics. Details of this design are available in [42, 43]. A technique to compare similarity of Galois lattice-based descriptor is also available from the same researchers. A technique to compose Galois lattice-based descriptor is available from Qi et al. [47]. This design assumes that concepts are only based on classification taxonomy. Based on this oversimplistic taxonomy it elaborately models all possible concepts present in the entire context (in other words it defines the entire context in terms of finite number of concepts). This is an unrealistic rigid model of a concept as used in human thought process.

Such elaborate rigid formal modeling and related computation does not deliver any substantial gains in terms of meaning comparison confidence and therefore it is too expensive in terms of computation. This design does not seem to be based on any particular model and current understanding from cognitive science. On the other hand the notion of requirement for parsimony in cognitive processing seems to disagree with the elaborate FCA-based modeling. Description of the entire context with all possible concepts may not be necessary. In addition, harvesting attribute and object relationship from the object text description as suggested in [43] is not a good solution because all the attribute–object relationships required to define the concept may not be available in the object text. Though a technique to compose Galois lattice is available, it does not elegantly support hierarchical concept composition in a manner that happens in human mind. Therefore Galois lattice design does not achieve anything substantial in terms of meaning representation or comparison, which simpler vector-or set-based models can-not.

Graph-based concept descriptor. This design is based on graph where each elementary concept is a node and the relationships between the elementary concepts are denoted as edges connecting those concepts. Details of this design are available in [44]. This design is based on the notion of conceptual structure in human mind as proposed by [48]. The technique to compose two elementary concepts to get a representation of the complex concept is also available from Anderson [44]. It might be possible to compare complex concepts based on available graph composition or mapping techniques like [49, 50]; however those techniques are fairly computation intensive and not very elegant.

Set-based object descriptors. The descriptor is a set of terms which are either picked up from the object (text) or assigned by human indexers. The query descriptor is either a similar set of terms or a Boolean (set) condition involving some terms (Fig. 5.6). The similarity value between object and query descriptors is always Boolean 0 (not similar) or 1 (similar). It is computed by either checking whether any of the query descriptor terms are present in the object descriptors or strictly verifying whether the given Boolean condition in the query is satisfied for the object descriptor (as illustrated in Fig. 5.6). For object O_1 in Fig. 5.6, we considered the stemmed (base) terms "sales," "manager," "receipt," "look," and "took" as the terms for indexing. We ignored the terms "the," "he," "at," "then," and "it" because they do not carry any useful information that may help to distinguish objects.

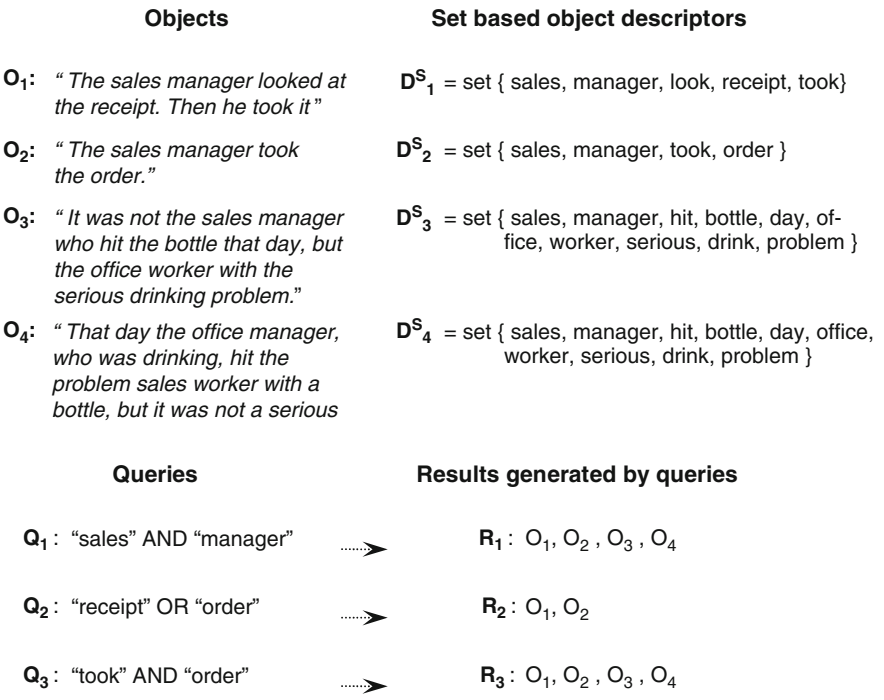


Fig. 5.6 Search operation with set-based descriptor [9]

Generally these terms that do not carry information occur with high frequency in all objects. Considering the set of four objects O_1, O_2, O_3, and O_4 as the collection, the three queries Q_1, Q_2, and Q_3 resulted in the three result sets R_1, R_2, and R_3. The objects O_3 and O_4 are borrowed from [51]. These two texts have similar terms but convey very different meanings.

Vector-based object descriptors. Here the descriptor of a text object is a vector where each dimension corresponds to a vocabulary term found in the object which should be included in the index. A term can be a single keyword, a compound word, or a phrase which indicates an entity or an idea [12]. If a term occurs in the descriptor, then the corresponding basis vector is assigned a non-zero scalar coefficient (term weight). The following example illustrates this idea.

We consider the text object "O_1" as in Fig. 5.6. The corresponding vector "V_1" for this text object consists of several basis vectors and is given as

$$V_1 = w_{sales,1} \cdot \overrightarrow{e}_{sales} + w_{manager,1} \cdot \overrightarrow{e}_{manager} + w_{receipt,1} \cdot \overrightarrow{e}_{receipt} + w_{look,1} \cdot \overrightarrow{e}_{look} + w_{took,1} \cdot \overrightarrow{e}_{took}$$

The basis vectors are shown as "e_i" with arrows over them and their corresponding scalar coefficients are indicated as "w_i". The descriptor "D_1" for this text object is the normalized vector "V_1" where the basis vectors remain the same, but their scalar coefficients are normalized. This descriptor is given by

$$D_1 = \frac{V_1}{\|V_1\|} = \frac{w_{sales,1} \cdot \overrightarrow{e}_{sales} + w_{manager,1} \cdot \overrightarrow{e}_{manager} + w_{receipt,1} \cdot \overrightarrow{e}_{receipt} + w_{look,1} \cdot \overrightarrow{e}_{look} + w_{took,1} \cdot \overrightarrow{e}_{took}}{\sqrt{(w_{sales,1})^2 + (w_{manager,1})^2 + (w_{receipt,1})^2 + (w_{look,1})^2 + (w_{took,1})^2}}$$

$$= x_{sales,1} \cdot \overrightarrow{e}_{sales} + x_{manager,1} \cdot \overrightarrow{e}_{manager} + x_{receipt,1} \cdot \overrightarrow{e}_{receipt} + x_{look,1} \cdot \overrightarrow{e}_{look} + x_{took,1} \cdot \overrightarrow{e}_{took}$$

where

$$x_i = \frac{w_i}{\sqrt{\sum i w_i^2}}$$

Here we assume that the terms "*sales,*" "*manager*," "*receipt*," "*look*," and "*took*" are not at all similar, therefore their basis vectors "e_{sales}," "e_{manager}," "e_{receipt}," "e_{look}," and "e_{took}" are orthogonal to each other. Which means the dot product, of these basis vector tensors, represented by $\langle \bullet, \bullet \rangle$, are all zero, i.e., $\langle e_{sales}, e_{manager} \rangle = \langle e_{manager}, e_{receipt} \rangle = \cdots = \langle e_{took}, e_{sales} \rangle = \cdots = 0$. Terms within the text that are similar to each other are handled in a different way as explained later.

Assignment of weights: A term which is an important distinguishing factor should have a higher weight. There are several alternative ways to assign these "w_i" values, the most popular being the term frequency–inverse document frequency (TF–IDF) scheme [12]. In this scheme a high weight is assigned to a term if it occurs frequently in a document but not frequently in all documents in the collection. The

weight "$w_{T,O}$" for a certain term "T" in the vector for a given object "O" in a collection is given by product of term frequency and inverse document frequency. This is computed by the following equation.

Semantic similarity between vector descriptors: The level of semantic similarity between two descriptor vectors, V_1 and V_2, is given by the cosine of the angle between the normalized vectors. This kind of comparison generates continuous values between 0 and 1. A similarity value of zero means that two vectors are dissimilar (orthogonal) and a higher value indicates more similarity.

$$\cos\theta = \langle D_1, D_2 \rangle = \left\langle \frac{V_1}{\|V_1\|}, \frac{V_1}{\|V_2\|} \right\rangle = \frac{V_1}{\|V_1\|} \bullet \frac{V_1}{\|V_2\|}$$

This is explained by the following example. Here we consider another text object "O_2," also from Fig. 5.6, having descriptor vector "D_2":

$$D_2 = x_{sales,2} \cdot \vec{e}_{sales} + x_{manager,2} \cdot \vec{e}_{manager} + x_{took,2} \cdot \vec{e}_{took} + x_{order,2} \cdot \vec{e}_{order}$$

The similarity between "D_1," which was presented earlier, and "D_2" is the sum of product of the scalar coefficients of all basis vectors that are common in the two vectors and then normalized. This is given by

$$D_1 \bullet D_2 = x_{sales,1} \cdot x_{sales,2} + x_{manager,1} \cdot x_{manager,2} + x_{took,1} \cdot x_{took,2}$$

Searching using vector-based descriptor: The search query is given as a string of terms or in the form of natural language text. A vector-based descriptor for the search string is generated and compared with the object descriptors using cosine similarity. The result objects are ranked based on the cosine similarity value. This is illustrated in Fig. 5.7 for the same objects used in Fig. 5.6 earlier. The collection of terms inside the quotes is treated as a single compound term and terms without quotation are treated as separate terms.

Limitations of vector models: Vector-based descriptor generation and comparisons are computation intensive. The TF–IDF computation assumes presence of

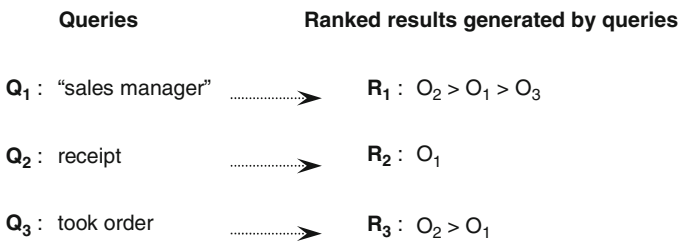


Fig. 5.7 Search operation with vector-based descriptor [9]

a centralized corpus. It also implies centralization of inverse document frequency computation and the index.

Vector-based approach also has an inherent weakness in representing complex descriptions which are based on complex meanings (concepts). This results in failure to discern between descriptions that have common keywords/terms but have very different meanings. This is well explained by objects O_3 and O_4 as in Fig. 5.6, which have similar terms but convey different meanings. A simple vector-based similarity computation erroneously reports that these objects are similar (the TF–IDF-based cosine similarity is 0.998). Considering compound words or named entities like "*sales manager*," "*office manager*," "*office worker*," "*sales worker*," or "*problem sales worker*" as the terms can improve the situation but will not report absolute zero similarity and thus cannot entirely avoid this problem in all situations. Instead of machine-generated vectors, if the vectors are based on standardized topics assigned by human indexers, then some of these problems can be avoided. However, this requires human involvement.

This problem exists primarily because vector-based models are based on a belief that elementary syntactic terms contain the meanings and this meaning can be entirely captured by a flat collection of the terms available in the text. Some of these limitations have caught attention of researchers and enhancements like vector products [51, 52] and more sophisticated vector operations like [53, 54] are being proposed. However, these are term-centric approaches which will have limited impact in meaning representation. We need new kind of designs for semantic descriptors which are truly high level meaning centric approaches.

Semantic similarity between terms. To extend the vector cosine product similarity approach to suit meaning-based search, there is a need to account for the similarity between meanings conveyed by different terms. Considering the example in Table 5.1, the object, query vectors, and their cosine product (with only non-orthogonal terms) would be

$$D_3 = x_{rodent,3} \cdot \overrightarrow{e}_{rodent} + x_{\text{sup}\,plier,3} \cdot \overrightarrow{e}_{\text{sup}\,plier,3} \text{ and } D_4 = x_{mouse,4} \cdot \overrightarrow{e}_{mouse} + x_{\text{sup}\,plier,4} \cdot \overrightarrow{e}_{\text{sup}\,plier}$$

$$D_3 \bullet D_4 = x_{rodent,3} \cdot x_{mouse,4} \cdot y + x_{\text{sup}\,plier,3} \cdot x_{\text{sup}\,plier,4}$$

Therefore there is need to recognize that the terms "rodent" and "mouse" are not dissimilar (non-orthogonal basis vector) and there is a non-zero semantic similarity value "y" which weights their dot products (which may not be 1 always). To handle these kinds of issues during semantic searching, there is need to identify or assign real values to the parameter "y" in all such pairs of terms (non-orthogonal basis vectors). Hence we need techniques to identify similarity of meaning between different terms in the first place. Various methods have been proposed to quantify semantic similarity between lexical terms (synonyms, hypernyms, co-occurring terms). Some of these are briefly discussed below.

Feature-based models: Here each term is modeled to have a set of attributes. For example "dog" has the following attributes: "pet," "mammal," "carnivore," "small

animal," etc. Considering these attributes as dimensions, the semantic similarity between two terms can be based on the cosine similarity measure or Euclidean distance. Other ways to calculate similarity are based on how many attributes two terms have in common. Several alternative ways to make these computations have been documented in [55, 56].

It is also important to choose a minimal set of dimensions which will correctly cluster terms according to their semantic distances. Automatic generation of these attributes (dimensions) and dimension reduction techniques like Vector Generation from Explicitly-defined Multi-dimensional semantic space (VGEM) [57]. Best path Length On a Semantic Self-organizing Map (BLOSSOM) [58], and Latent Semantic Analysis (LSA) [59] are useful. After clustering, the similarity value is assigned based on distances between terms.

Ontology/taxonomy-based models: To compute similarity between individual terms, these methods consider how far away is one term from another in a global ontology or taxonomy graph (e.g., Fig. 5.8) in terms of number of edges that need to be traversed to reach one term node from another; how many common ancestors they share; or how far away is the lowest level common ancestor, etc. [60–63]. Another method is to calculate how much common information the two terms share

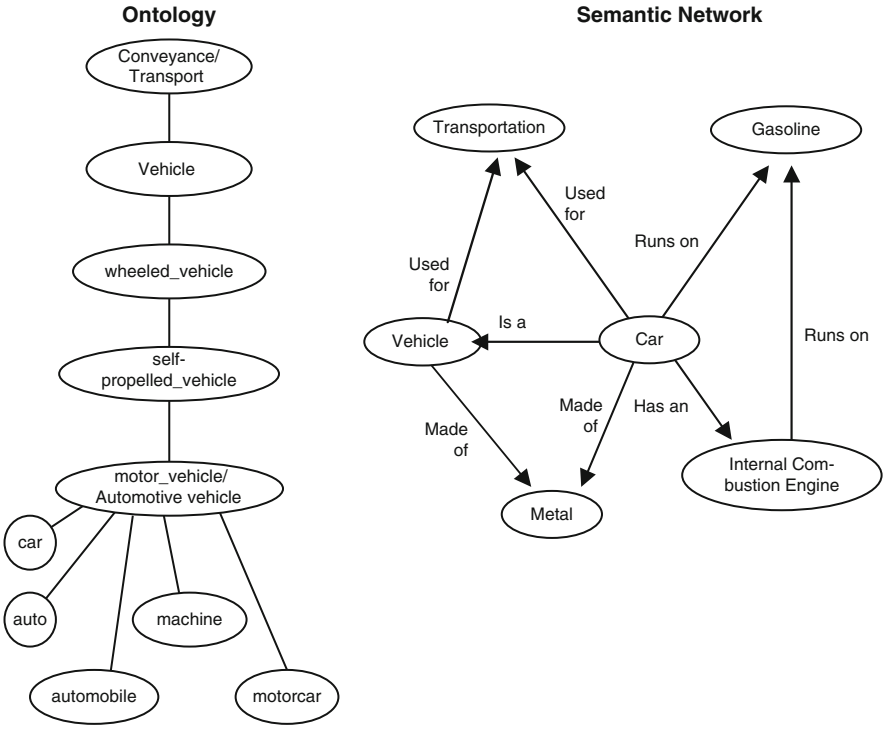


Fig. 5.8 Example of ontology and semantic network with distance between two terms [9]

through a concept/term that subsumes both terms. The information content is given by negative log of the reciprocal of the frequency of occurrences of a subsuming term in the corpus (probability of occurrence) [56].

Corpus and semantic network-based models: Here the assumption is that more information two terms share, more similar they are. The amount of information they share is given by the concurrence probability. Normalized Google distance is a classic example [64], where the similarity distance between two terms T_1 and T_2 is given by

$$\mathrm{sim}(T_1, T_2) = \frac{\max\left(\log(number\_of\_objects\_having\_T_1), \log(number\_of\_objects\_having\_T_2)\right) - \log(number\_of\_objects\_having\_both\_T_1\_\&\_T_2)}{Total\_number\_of\_objects - \min(\log(number\_of\_objects\_having\_T_1), \log(number\_of\_objects\_having\_T_2)}$$

Semantic networks (e.g., Fig. 5.8) generated from a corpus can be also used to identify the semantic similarity between terms. In these methods the similarity is based on how far away one term is from another in the semantic network [65–67].

Description logic-based models: These approaches [68–71] depend on explicit logical descriptions of terms often using RDF [72] expression and identify similarity based on inductive inferences. These are some basic techniques behind semantic web proposals.

Word sense disambiguation of terms based on context is another associated problem which has to be addressed. In the example as given in the second row in Table 5.1, if the user had been searching for computer mouse, then considering "y" > 0 (refer previous section) would not be proper. So there is a need to disambiguate the term "mouse" appropriately to choose the right value of "y" to use given the context. Terms which need disambiguation can be detected using Explicit Semantic Analysis (ESA) [73] and analyzing graph patterns in the semantic network [14] constructed from corpus and then noting the pattern of connections [74].

5.5 Proposed Semantic Technologies for Cyber-Infrastructure

Here we propose a design of semantic descriptor and related techniques. In this proposed method, to make an object (text document or service) searchable by intention-based web search, the owner of the object must properly publish it on the web (Fig. 5.9).

To publish the object it has to be placed in the web server and a text description of the object has to be provided to the indexing mechanism. The index mechanism will generate semantic descriptor (key) of the object and submit the key object address pair to the index system (which maintains key to address mappings), whereas during intention-based searching a user has to provide a text description of the desired object to the search service's user interface. The user interface generates a semantic descriptor of the intention and uses it to execute the search. Both of these processes require a method to generate semantic descriptor from the given natural language text description. The subsequent sections explain this method.

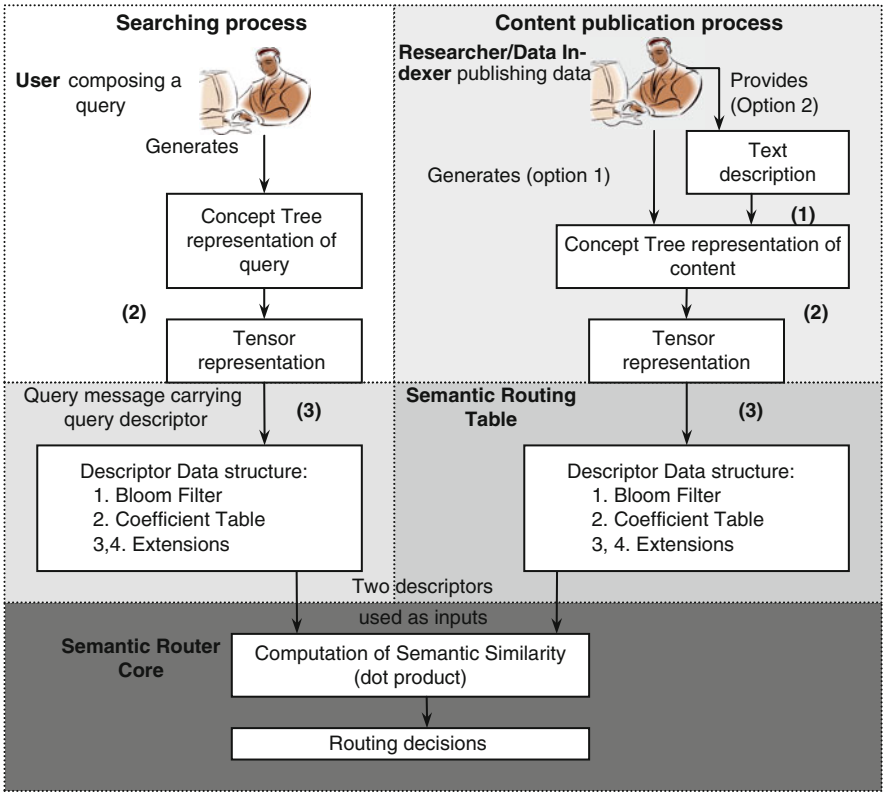


Fig. 5.9 Overview of descriptor related-processes [9]

5.5.1 Overview

A technique to generate semantic descriptor from the text description of an intention or an object is presented in Fig. 5.10 [9, 45]. The generation process consists of three steps. Bloom filter-based data structure is the final output that is generated from the text representation.

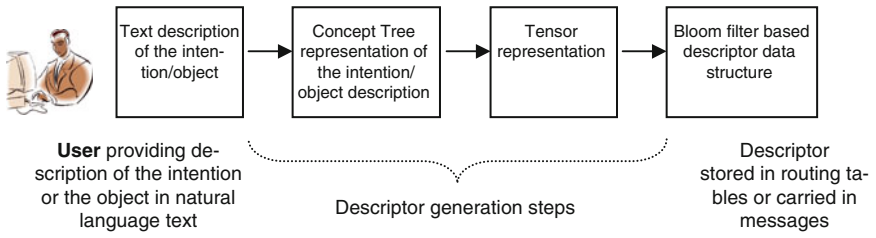


Fig. 5.10 Descriptor generation process [9]

In the first step, the meaning of the text, which is a complex concept, is captured and represented as concept tree which has basic concepts as its leaves. Basic concepts are the ones which are defined by standard terms in the domain lexicon or ontology. In the second step an algebraic (tensor) representation of this concept tree is constructed and in the third step this derived tensor is encoded in a Bloom filter-based data structure. The algebraic representation of the tree enables the use of cosine similarity comparison method to compare two trees. The bloom filter-based data structure for the algebraic representation allows us to carry out the cosine similarity as parallel computation on a specialized but simple hardware. This reduces processing time drastically, which is necessary for fast descriptor comparison and semantic routing. These three processing steps and four representations are explained with an example in the following sections.

5.5.2 Concept Tree Representation

Generation process. To explain the descriptor generation process [9] we have considered a bioscience publication [75] as an object and its abstract as the textual description. Figure 5.11 presents this text description along with its corresponding concept tree representation. In the concept tree figure, the composed concepts are underlined and the basic concepts (terms) are shown in bold. The basic concepts are used as the leaves. The basic concepts are the ones which have been defined as controlled terms in domain ontologies like Gene Ontology [76] and Disease ontology [77].

Here the given text is a publication considered as a narrative about the disease diabetes and the implicated gene 1858T and its normal variant 1858C. Therefore the entire publication is expressed as a concept, which is shown at the top of the tree. The concept is composed of two child concepts: the resulting disease and the gene. Here each concept is defined by collection of elementary concepts. The rationale behind this representation is explained later.

The next two levels of the tree describe the gene in terms of the parent gene (PTPN22) and its function, which is encoding of the associated protein (Lyp). The concept of the "Lyp" protein is defined in multiple ways: by its name "LyP"; by its "binds with" relationship with another gene product "Csk"; and by its function "negative regulation of T-cell activation." Thus the "Lyp" concept is represented by a collection of all these elementary concepts.

Construction rules. The specific rules of constructing this tree will depend on the domain knowledge models and these rules can be codified as possible composition templates to be used to represent concepts as and when required. Here three composition templates were used. The first one was the composition of disease name and implicated gene which served as the template to describe the publication. The second one was a composition of gene name, protein name, and other gene attributes (variations), which was used to represent the gene "PTPN22." The third template was a composition of protein name and functions, which was used to represent the protein "Lyp." These standard templates can be put in a library to

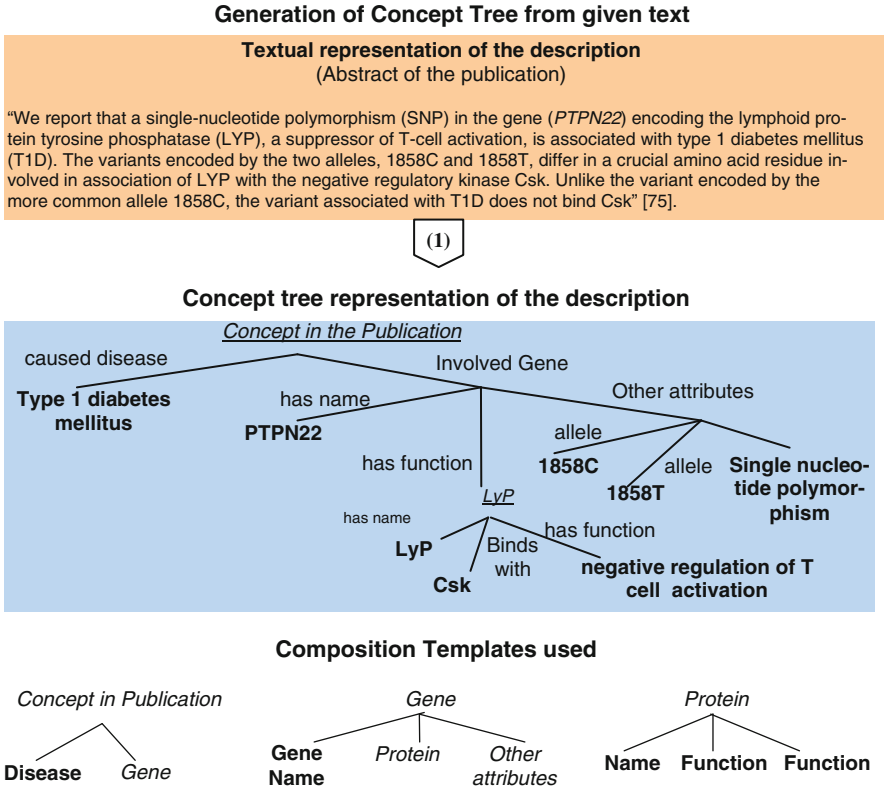


Fig. 5.11 Concept tree generation from text narrative [9]

be used during construction of the concept tree. Using these standard templates is important because only that way all keys will be generated using a common set of rules to construct concept trees. This will ensure that similar concepts are represented by similar concept tree representations (descriptors). This will enable proper similarity comparison between meanings.

The keywords or phrases used at the leaves are selected from a controlled vocabulary (based on the text) to represent the meaning of the text for indexing purpose. When sufficient controlled vocabulary is available for a given domain, this technique can be applied to generate concept trees from text. The actual nature of relationships between concepts is ignored (however shown in Fig. 5.11 for the purpose of illustration only).

Rationale. The justification of concept tree representation is based on the notions presented in Section 5.4.3. There we discussed how complex meaning can be represented in the form of a tree structure where the nodes at the intermediate levels are hierarchical collection of elementary meanings. The equivalence between concept tree and hierarchy of collections of elementary meanings is illustrated in Fig. 5.12.

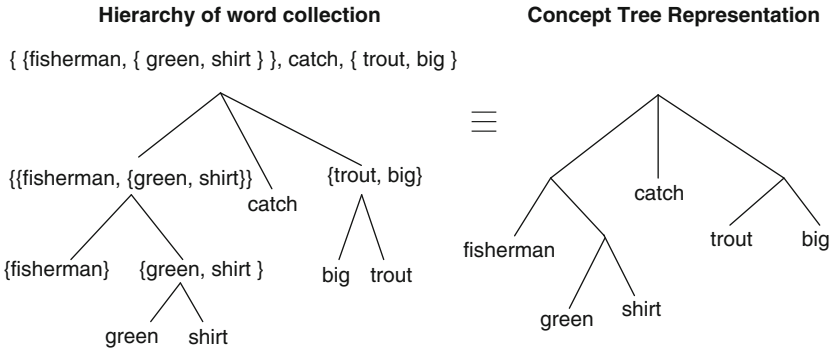


Fig. 5.12 Equivalence between hierarchy of collections of elements and concept tree [9]

This shows that a concept tree can represent a complex concept (meaning) and has backing from cognitive science. Therefore this satisfies requirement (1) and (2).

Neuro-scientific findings reported in [28] indicate that human brain uses similar neurological processes to comprehend meaning from text and visual imagery. Other evidences [31, 32] suggested that semantic processing takes place at a common site irrespective of the kind of language that is being processed. This corroborates well with the argument that semantic processing and composition has its own set of rules. It seems that the semantic processing and composition rules are perhaps common across different languages including visual imagery. As the concept tree is based on the minimal set of semantic composition rules, and as these same set of rules are also the basis behind all these languages [28, 35], therefore concept tree is compatible with all these languages (and visual imagery). Hence seamless transfer of meaning from these languages (texts and images) to concept tree representations is possible. Therefore concept tree representation satisfies requirement (3).

5.5.3 Required Algebra

Definition of the vector space. An idea or concept is expressed as a tensor [78] in an infinite-dimensional space. This space is represented by two kinds of basis vectors. One kind comprises of a set of basic basis vectors "e_i," each of which corresponds to a unique basic concept (e.g., "Csk," "LyP," "negative regulation of T-cell activation") in the domain lexicon. The other kind includes basis vectors which are polyadic combinations represented in the form "$e_i e_j e_k \ldots$ ect." which represent conjunction of concepts (e.g., "LyP" and "CsK" and "negative regulation of T-cell activation," and ect.). This second kind of basis vector is needed because elementary concepts can combine with each other to form complex concepts which are entirely different from the elementary ones. The basis vectors and their polyads are hence orthogonal.

Algebraic representation of composition. An ordered composition of three tensors: A, B, and C can be represented as a triadic [78] tensor product ABC which is represented as an ordered juxtaposition of three individual tensors. The tensor product ABC represents a conjunction composition where elementary tensors A, B, and C are arranged in a specific order from left to right. Similarly a conjunction composition with any arbitrary number of elements can be represented by a polyadic tensor product [9, 45]. However this simple tensor product is not sufficient to represent a tree because it does not satisfy the two important requirements:

1. Do not enforce ordering of children nodes.
2. Represent both conjunction and disjunction composition at the same time.

We need a different algebraic representation that can satisfy both these requirements to represent a tree, which does not enforce ordering of elements. To achieve this purpose we introduce two algebraic binders (functions) (1) [•, •, ..] and (2) {•, •, ...}, that bind two or more tensors (concepts) together. For example, by using the binder function [•, •, ..], three tensors A, B, and C can be bound together to represent a composition [A,B,C]. Using these two binder functions we will synthesize an algebraic (tensor) representation that can depict a concept tree in terms of its leaves. These binder functions represent compositions which do not enforce ordering of arguments. This means that these binder functions are commutative with respect to their arguments. This ensures that all possible isomorphic trees (Fig. 5.13) that convey the same meaning are expressed by a single tensor.

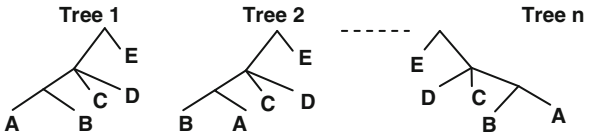


Fig. 5.13 Isomorphic concept trees which convey the same meaning [45]

Definition of [•, •, ..] *binder*. For the case of one, two, and three arguments we define as follows:

AB denotes a dyadic tensor product, ABC denotes a triadic tensor, and a polyadic tensor is denoted by a juxtaposition (e.g., ABCD). In general, AB ≠ BA. This definition can be expanded for a general case of "*n*" arguments, where the sum of product form has all permutations of arguments: A,B,C, etc. Figure 5.14 illustrates the usefulness of this binder.

Role of [•, •, ..] *binder*: This binder represents a conjunction composition that does not impose ordering of leaves.

Proof for the commutative property: Proof for two argument cases is given which can be extended for "*n*" arguments. [A, B] = AB +BA = BA + AB = [B, A].

Definition of {•, •, ...} *binder*. For one, two and three arguments we define

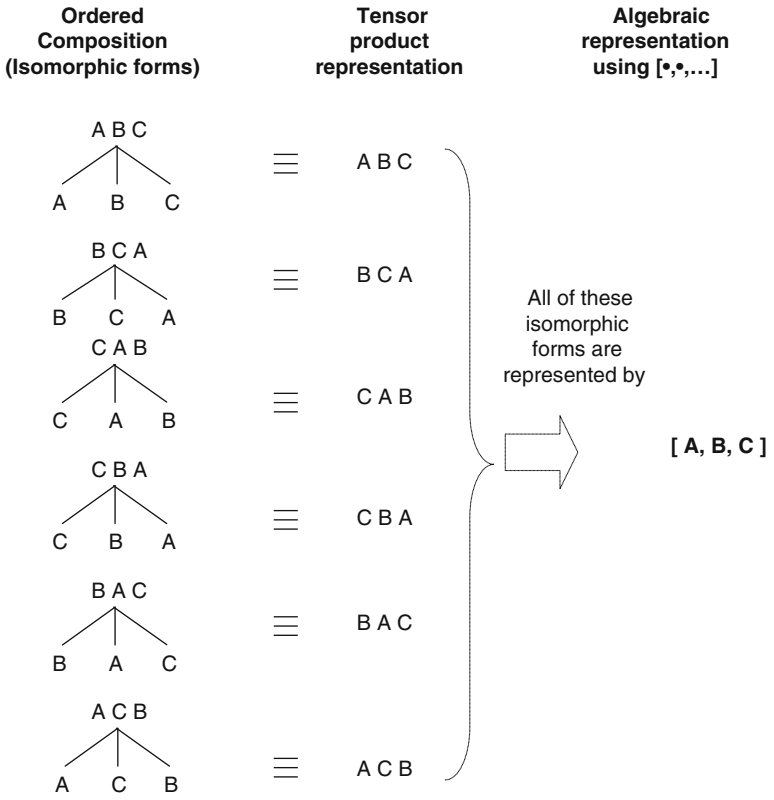


Fig. 5.14 Usefulness of [•, •, ..] binder [9]

$$\{A\} \equiv \frac{\sqrt{h_A}*[A]}{\sqrt{h_A^2}} = A; \{A,B\} \equiv \frac{\sqrt{h_{AB}/2}*[A,B]+\sqrt{h_A}*[A]+\sqrt{h_B}*[B]}{\left\|\sqrt{h_{AB}/2}*[A,B]+\sqrt{h_A}*[A]+\sqrt{h_B}*[B]\right\|}$$

$$\{A,B,C\} \equiv \frac{(\sqrt{h_{ABC}/6}*[A,B,C]+\sqrt{h_{AB}/2}*[A,B]+\sqrt{h_{BC}/2}*[B,C]+\sqrt{h_{AC}/2}*[A,C]+\sqrt{h_A}*[A]+\sqrt{h_B}*[B]+\sqrt{h_C}*[C])}{\left\|(\sqrt{h_{ABC}/6}*[A,B,C]+\sqrt{h_{AB}/2}*[A,B]+\sqrt{h_{BC}/2}*[B,C]+\sqrt{h_{AC}/2}*[A,C]+\sqrt{h_A}*[A]+\sqrt{h_B}*[B]+\sqrt{h_C}*[C])\right\|}$$

This binder encompasses all possible combinations and permutations of arguments. The resultant tensor is also normalized and used as an elementary tensor to be incorporated for next higher level of composition. Each instance of this binder has a corresponding set of co-occurring coefficients "H," having real-valued scalar elements (e.g., H = set $\{h_{\mathrm{ABC}}, h_{\mathrm{AB}}, h_{\mathrm{BC}}, h_{\mathrm{AC}}, h_{\mathrm{A}}, h_{\mathrm{B}}, h_{\mathrm{C}}\}$), each of which indicates the importance of the corresponding polyad to represent the meaning of the composed concept. For example, when only $h_{\mathrm{ABC}} = 1$ and all other scalars $h_{\mathrm{AB}} = h_{\mathrm{BC}} \ldots = h_{\mathrm{C}} = 0$, then the composed concept is the one which is given by a strict conjunction of A,B, and C, whereas the set $h_{\mathrm{A}} = h_{\mathrm{B}} = h_{\mathrm{C}} = 1$ and $h_{\mathrm{ABC}} = h_{\mathrm{AB}} = h_{\mathrm{BC}} = h_{\mathrm{AC}} = 0$ represents disjunction composition. A mix of all these extremes is

possible by suitable choice of values for the co-occurring coefficients. Rules that guide assignment of these values can be codified and made accessible along with composition templates. These parameters are normalized by $(n!)^{1/2}$, where "n" is the number of arguments in {•, •, ...} binder.

Proof for the commutative property: When a function "F" is a linear composition (linear functional) of two other functions "F_1" and "F_2," such that $F = \lambda_1 {}^*F_1 + \lambda_2 {}^*F_2$ where λ_1 and λ_2 are real numbers, and if both functions F_1 and F_2 are commutative, then their linear functional F is also commutative. The proof of this property is shown for two arguments, but it can be extended for "n" arguments.

$$\text{If } F_1(A,B) = F_1(B,A) \text{ and } F_2(A,B) = F_2(B,A),$$

$$\text{Therefore } F(A,B) = \lambda_1 {}^*F_1(A,B) + \lambda_2 {}^*F_2(A,B) = \lambda_1 {}^*F_1(B,A) + \lambda_2 {}^*F_2(B,A) = F(B,A)$$

As binder {•, •, ...} is a linear functional of commutative [•, •, ..] binders, binder {•, •, ...} is also commutative.

5.5.4 Tensor Representation of Concept Tree

A concept tree can be represented by a tensor which is expressed as a sum of scalar-weighted polyads of the basic basis vectors that represent the leaves. We illustrate this with examples in Figs. 5.15 and 5.16, where each tree has three levels. A tensor representation consists of basis vectors and their scalar coefficients. First we show how to generate the basis vectors for the tensor that represents the entire concept tree. Then we generate the scalar coefficients for each of these basis vectors.

Generation of basis vectors: The trees and the forms of their tensor representations are shown in Fig. 5.15. Basic basis vectors are denoted by lower case

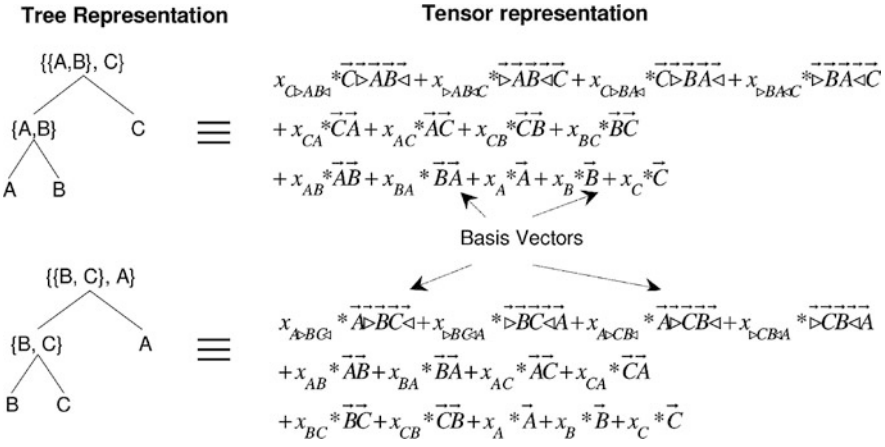


Fig. 5.15 Basis vectors for the tensor for a simple three-level concept tree [9]

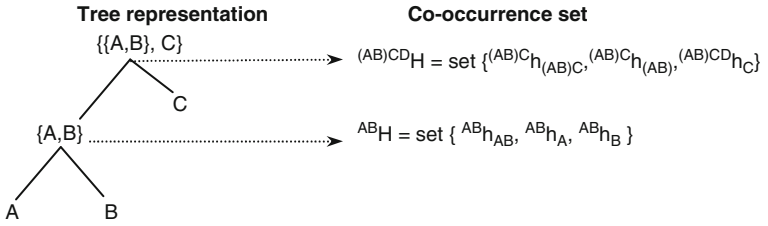


Fig. 5.16 Tensor expression for a concept tree [9, 45]

alphabets with arrow on top and scalar coefficients are without arrows. Delimiter vectors "▷" and "◁" are introduced between tensors which are at different branches (compositions). The delimiter vectors point toward the tensor which belongs to another branch. For example, instead of "CAB" and "ABC" we write "C▷AB◁" and "▷AB◁C." The use of delimiter vectors ensures that trees having same leaves but different composition do not have similarity beyond which is contributed by the individual leaves (Fig. 5.15). The ordering and combination of the leaf tensors and the delimiter vectors "▷" and "◁" in the polyadic products retain the information about the tree structure [9].

Generation of scalar coefficients: In Fig. 5.16 all the intermediate composed concepts for this tree are represented by the expressions present at corresponding node positions in the concept tree. These expressions are represented in terms of the binders defined earlier. On expanding these expressions we get the tensor in the form which is suitable for inner product (cosine similarity) computation. Each composition in this tree has its own co-occurrence set, for example, the set "H" for binder {A,B} is denoted by ${}^{\mathrm{AB}}H$. For two compositions in example tree, the two co-occurrence sets ${}^{\mathrm{AB}}H$, ${}^{(\mathrm{AB})\mathrm{C}}H$ are also shown by arrows in Fig. 5.16.

The expression {{A,B}, C} which represents the tree in Fig. 5.16 is expanded bottom-up as an example using the basis vectors identified earlier.

The scalar coefficients of the basis vectors are products of normalized co-occurring coefficients. The basic basis vectors are shown here as single alphabets: a, b, c, etc., with arrows on top of them, and their polyads are shown as juxtaposition of these alphabets.

$$A = \overrightarrow{a}, B = \overrightarrow{b}, C = \overrightarrow{c}, \text{ and } \{A, B\} = \frac{{}^{AB}h_{AB} \Big/ \sqrt{2} * [A, B] + {}^{AB}h_A * [A] + {}^{AB}h_B * [B]}{\left\| {}^{AB}h_{AB} \Big/ \sqrt{2} * [A, B] + {}^{AB}h_A * [A] + {}^{AB}h_B * [B] \right\|}$$

$$= \frac{{}^{AB}h_{AB} \Big/ \sqrt{2}(\overrightarrow{a}\,\overrightarrow{b} + \overrightarrow{b}\,\overrightarrow{a}) + {}^{AB}h_A\,\overrightarrow{a} + {}^{AB}h_B\,\overrightarrow{b}}{\sqrt{({}^{AB}h_{AB})^2 + ({}^{AB}h_A)^2 + ({}^{AB}h_B)^2}}$$

$$\{\{A,B\},C\}$$

$$= \frac{{}^{(AB)C}h_{(AB)C}\Big/\sqrt{2} * \triangleright \{A,B\} \triangleleft C + {}^{(AB)C}h_{(AB)C}\Big/\sqrt{2} * C \triangleright \{A,B\} \triangleleft + {}^{(AB)C}h_{AB} * \{A,B\} + {}^{(AB)C}h_C * C}{\sqrt{({}^{(AB)C}h_{(AB)C})^2 + ({}^{(AB)C}h_{AB})^2 + ({}^{(AB)C}h_C)^2}}$$

$$= \frac{1}{\sqrt{({}^{AB}h_{AB})^2 + ({}^{AB}h_A)^2 + ({}^{AB}h_B)^2}} * \frac{1}{\sqrt{({}^{(AB)C}h_{(AB)C})^2 + ({}^{(AB)C}h_{AB})^2 + ({}^{(AB)C}h_C)^2}} *$$

$$\left(\frac{{}^{(AB)C}h_{(AB)C}}{\sqrt{2}} \frac{{}^{AB}h_{AB}}{\sqrt{2}} * (\overrightarrow{\triangleright}\ \overrightarrow{a}\ \overrightarrow{b}\ \overrightarrow{\triangleleft}\ \overrightarrow{c} + \overrightarrow{\triangleright}\ \overrightarrow{b}\ \overrightarrow{a}\ \overrightarrow{\triangleleft}\ \overrightarrow{c} + \overrightarrow{c}\ \overrightarrow{\triangleright}\ \overrightarrow{a}\ \overrightarrow{b}\ \overrightarrow{\triangleleft} + \overrightarrow{c}\ \overrightarrow{\triangleright}\ \overrightarrow{b}\ \overrightarrow{a}\ \overrightarrow{\triangleleft} \right)$$

$$+ \frac{{}^{(AB)C}h_{(AB)C}}{\sqrt{2}} * {}^{AB}h_A * (\overrightarrow{a}\ \overrightarrow{c} + \overrightarrow{c}\ \overrightarrow{a})$$

$$+ \frac{{}^{(AB)C}h_{(AB)C}}{\sqrt{2}} * {}^{AB}h_B * (\overrightarrow{b}\ \overrightarrow{c} + \overrightarrow{c}\ \overrightarrow{b}) + {}^{(AB)C}h_{AB} * \frac{{}^{AB}h_B}{\sqrt{2}} * (\overrightarrow{a}\ \overrightarrow{b} + \overrightarrow{b}\ \overrightarrow{a})$$

$$+ {}^{(AB)C}h_{AB} * {}^{AB}h_A * \overrightarrow{a} + {}^{(AB)C}h_{AB} * {}^{AB}h_B * \overrightarrow{b} + {}^{(AB)C}h_C * \overrightarrow{c})$$

Superior performance of tensor-based approach: The following example illustrates how tensor model works. Here we compared the performance of our tensor-based semantic similarity computation scheme against the vector-based approach. We obtained four publications (objects) from PubMed [1] on gene–diabetes interaction studies, which are denoted by O_i in Table 5.3. The object pairs are ranked based on (1) human interpretation; (2) semantic similarity values obtained by the tensor scheme; and (3) similarity values given by the vector-based approach. MeSH index terms and their hypernyms were considered as the vectors in the vector model as in case of exploded search [13]. Semantic similarity ranking is our comparison criterion here, since it is the key primitive operation carried out during semantic lookup and semantic routing table optimization [17].

The non-parametric Kendall tau correlation of the rankings based on tensor and human interpretation is 0.867 which is much higher than that between vector approach and human interpretation (which is 0.067). This shows that the tensor-based semantic descriptor model agrees more closely with humans in meaning comparison.

Table 5.3 Superior performance of tensor-based approach for object similarity rankings [9, 45]

	Semantic similarity rankings and (similarity values)		
Object pairs	Human	Tensor approach	Vector approach
O_1, O_2	Rank 1	Rank 1 (0.864)	Rank 4 (0.442)
O_3, O_4	Rank 2	Rank 2 (0.689)	Rank 1 (0.653)
O_2, O_3	Rank 3	Rank 3 (0.557)	Rank 5 (0.395)
O_1, O_3	Rank 4	Rank 5 (0.443)	Rank 3 (0.521)
O_2, O_4	Rank 5	Rank 4 (0.525)	Rank 6 (0.376)
O_1, O_4	Rank 6	Rank 6 (0.317)	Rank 2 (0.608)
Kendall tau rank correlation	1	0.867	0.067

5.5.5 Bloom Filter Basics

The inner (dot) product $\langle A, B \rangle$ of two concept tensors A and B is the sum of product of the scalar coefficients of all basis vectors that are common in A and B. To quickly pair up the scalar coefficients for multiplications we use bloom filters as explained below.

$$A = s_i^A \overrightarrow{i} + s_j^A \overrightarrow{j} + s_k^A \overrightarrow{k} + s_l^A \overrightarrow{l} + s_m^A \overrightarrow{m} + s_n^A \overrightarrow{n}; B = s_i^B \overrightarrow{i} + s_p^B \overrightarrow{p} + s_q^B \overrightarrow{q} + s_k^B \overrightarrow{k}$$
$$\langle A, B \rangle = s_i^A s_i^B + s_k^A s_k^B$$

Bloom filter (BF) is a compact representation of a set [79, 80]. A BF is a large single dimensional array of "m" bits and a set of "k" hash functions (Fig. 5.17).

To insert an element (a number or a text string) in this set we hash this element to generate "k" different values using the "k" different hash functions. We use these values as bit indices to decide which bits in the array should be set to 1. To test whether an arbitrary element is in the BF, we similarly generate "k" bit indices and check whether all of those bits are 1 or not. All bits being 1 indicates that the element is in the set (Fig. 5.17).

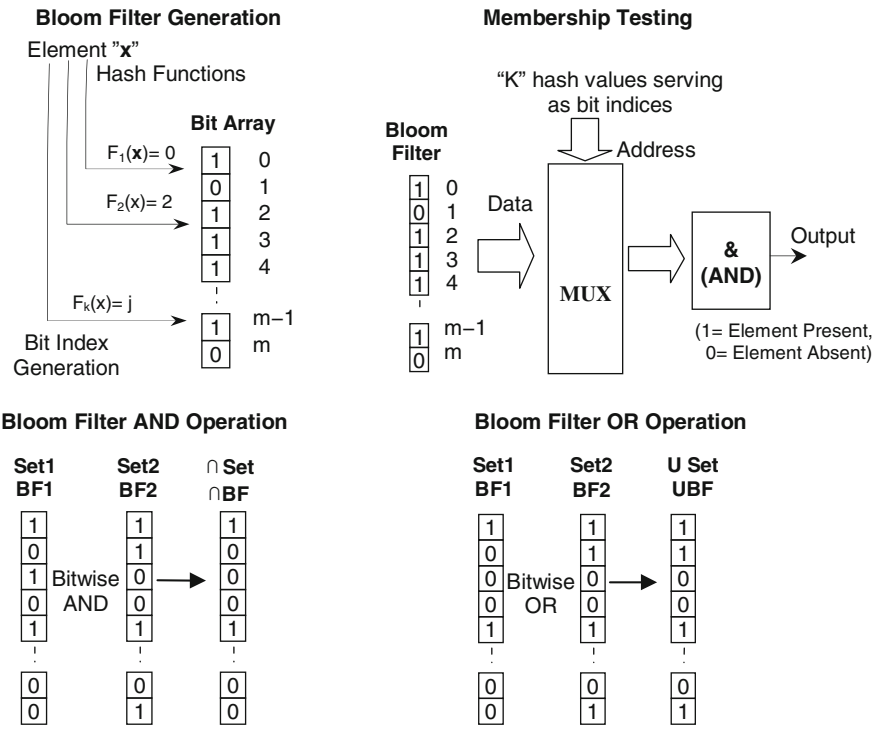


Fig. 5.17 Bloom filter basics: element insertion, membership testing, intersection, and union [9]

The test of whether an arbitrary element is in the BF can result in false positives (returns true even when the element is not present), but not false negatives (will never return false when the element is present). The probability of false positives, $p_{\text{false+ve}} \approx (1 - e^{-kn/m})^k$, can be minimized by choosing large "m," and optimum "k" ($\approx 0.7^*m/n$), where "m" is the number of bits in the BF, "k" is the number of independent hash functions, and "n" is number of elements in the BF (Fig. 5.17).

For example, a basic BF with $m = 132$ kbits, $k = 9$ can keep 10^4 elements with a small $p_{\text{false+ve}} \approx 2^*10^{-3}$, which will have a negligible effect on similarity comparison computation. When two similar sized BFs (same "m," "k") represent two sets then the BF obtained by bitwise AND operation of these two parent BFs (BF_1, BF_2 in Fig. 5.17) represents a set which is an intersection of the two sets. When tensor/vectors' basis vectors are inserted in two BFs, then the intersection BF is a set of common basis vectors whose scalar coefficients should be paired up for multiplication. This technique is explained in detail later. Similarly bitwise OR operation generates a BF which represents a set that is the union of the two parent sets (Fig. 5.17).

5.5.6 Generation of Bloom Filter-Based Descriptor Data Structure

The tensor representation of the entire concept tree is encoded in a compact semantic descriptor data structure, which has two basic components: (1) a big "m"-bit-wide bloom filter BF using "k" hash functions and (2) a coefficient lookup table (Fig. 5.18). Each element/row in the coefficient table has three columns: the 128-bit id for a basis vector (vector id); the scalar coefficient of the basis vector, as explained earlier; and a set of "k" $\log_2 m$ bit integers which are indexes for the BF bits that should be set to 1 when the basis vector id is inserted in the BF. These data structure components together will be either transmitted as query key, stored as semantic routing table row keys, or used to compute semantic similarity. The steps to encode the concept tree tensor in this data structure are explained below:

- **Step 1:** The 128-bit vector identifiers for the basis vectors (polyads) are generated by hashing the concatenated textual representation of the polyad by MD5 algorithm. For example, one of the polyad in the tensor for the tree will be a combination of five basic terms/vectors: "LyP"; "Csk"; "negative regulation of T-cell activation"; "▷"; "◁"; and "PTPN22." We concatenate these three character strings together to get "▷LyPCsk negative regulation of T-cell activation◁PTPN22" and then get its 128-bit MD5 digest. Henceforth we use the term vector ids and vectors interchangeably.
- **Step 2:** We insert the vector ids in the BF and note its BF bit cell indices.
- **Step 3:** For each basis vector, we insert a row in the coefficient lookup table comprising of the vector id, the scalar coefficient of that vector, and the set of all the BF bit cell indices (Fig. 5.18).

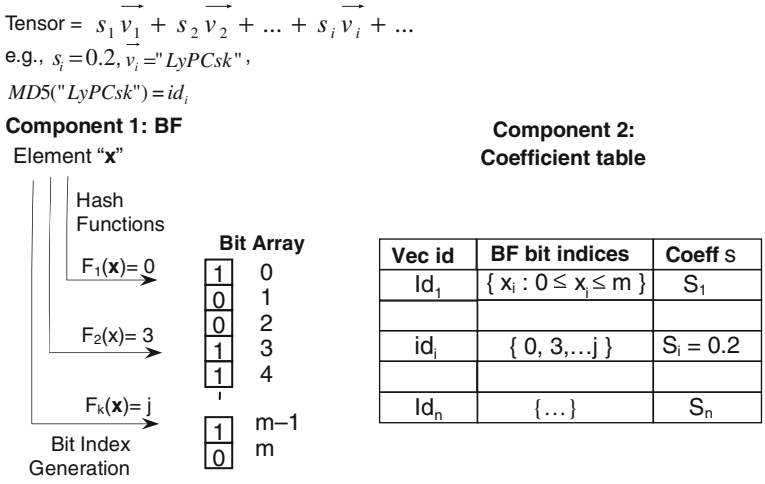


Fig. 5.18 Semantic descriptor data structure generation [9, 45]

5.5.7 Descriptor Comparison Algorithm

The steps for computing the inner product of two descriptors (D_1, D_2, Fig. 5.19) are as follows:

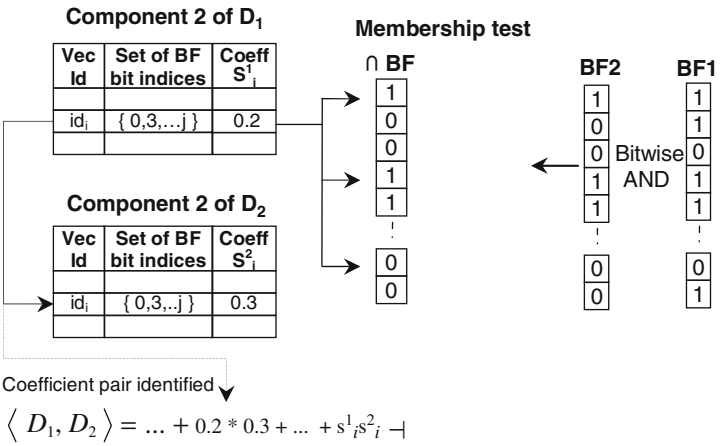


Fig. 5.19 Semantic descriptor comparison [9, 45]

- **Step 1:** Two BFs of the data structures being compared are intersected (ANDed).
- **Step 2:** Once the set of common basis vectors has been filtered out in step 1, these are identified from the coefficient lookup table by verifying which vector ids are in the intersection BF by using the set of the BF bit indices.

- **Step 3:** If a vector is present in the intersection BF, then we use that common vector id as the key and extract the coefficient value from the coefficient lookup table of the other data structure.
- **Step 4:** We multiply the pair of the coefficients for each identified common basis vectors and add all the products to get the similarity metric.

Each of these operations in steps 1 to 4 can be executed in parallel at high speeds on simple hardware accelerators. When there are "n" basis vectors in the smallest descriptor (tensor), "m" number of bits in the BF bit array, and "k" number of hash functions, which is generally $\leq$24, then a simple hardware accelerator can complete this computation in $(c + k)$ amount of clock cycles, where "c" is a constant $\approx$ 100. The order of memory and hardware complexity is O(n^*m).

5.5.8 Extensions for Incorporating Synonym and Hypernyms

To incorporate synonyms and hypernyms for the elementary terms, the basic data structure described in Section 5.5.6 is extended by a third and fourth component: (3) a collection (set) of synonymous vector sets and (4) a set of hypernym sequences.

Extensions for synonym. In the example of Fig. 5.20 the concept PTPN22 is denoted by multiple synonyms like "IPI00298016," "PTPN8: Tyrosine–protein phosphatase non-receptor type 22," in addition to "PTPN22," so there is a need to extend the concept tree tensor to incorporate all the known synonyms to enable correct comparison when two descriptors are constructed using two different synonyms. The required technique is explained with an abstract example (Fig. 5.20).

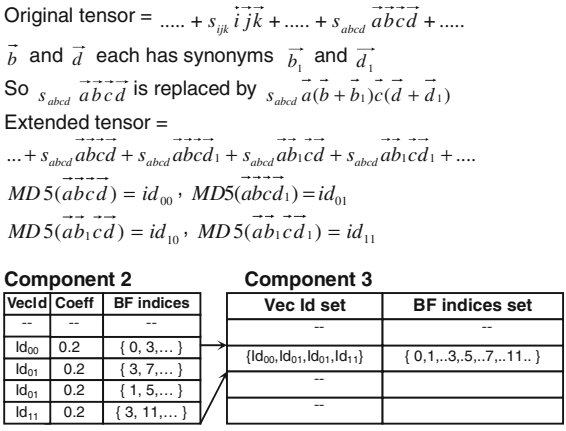


Fig. 5.20 Incorporation of synonyms in descriptor generation [9, 45]

Suppose the final tensor expression contains a term $s_{abcd}\,\vec{a}\,\vec{b}\,\vec{c}\,\vec{d}$, where the basis vector $\vec{a}\,\vec{b}\,\vec{c}\,\vec{d}$ is a polyad and the scalar coefficient is s_{abcd}. If $\vec{b}$ and $\vec{d}$

each have one synonym $\overrightarrow{b_1}$ and $\overrightarrow{d_1}$, then the term $s_{abcd}\,\overrightarrow{a}\,\overrightarrow{b}\,\overrightarrow{c}\,\overrightarrow{d}$ is replaced by the sum of all combinations of the synonyms: $s_{abcd}\,\overrightarrow{a}\,\overrightarrow{b}\,\overrightarrow{c}\,\overrightarrow{d} + s_{abcd}\,\overrightarrow{a}\,\overrightarrow{b}\,\overrightarrow{c}\,\overrightarrow{d_1} + s_{abcd}\,\overrightarrow{a}\,\overrightarrow{b_1}\,\overrightarrow{c}\,\overrightarrow{d} + s_{abcd}\,\overrightarrow{a}\,\overrightarrow{b_1}\,\overrightarrow{c}\,\overrightarrow{d_1}$. We call $\overrightarrow{a}\,\overrightarrow{b}\,\overrightarrow{c}\,\overrightarrow{d_1}$, $\overrightarrow{a}\,\overrightarrow{b_1}\,\overrightarrow{c}\,\overrightarrow{d}$, $\overrightarrow{a}\,\overrightarrow{b_1}\,\overrightarrow{c}\,\overrightarrow{d_1}$, and $\overrightarrow{a}\,\overrightarrow{b}\,\overrightarrow{c}\,\overrightarrow{d}$ as synonymous vectors. This can be generalized to incorporate any number of synonyms for any number of basic concepts. The data structure generating algorithm in Section 5.5.6 is extended as follows:

- **Steps 4 and 5:** The 128-bit vector ids for all the synonymous vectors are generated and inserted in the BF as in steps 1 and 2.
- **Step 6:** For each synonymous vector, we insert a row in the coefficient lookup table comprising the vector id, the scalar coefficient s_{abcd}, and the set of all the BF bit cell indices (Fig. 5.20).
- **Step 7:** The set of synonymous vector ids is grouped together as a set and paired with the set of all their BF bit indices, and inserted as an element in the third component, the "set of synonymous vector sets (SVS)" (Fig. 5.20).

A step 2a is inserted between steps 2 and 3 of the inner product algorithm in Section 5.5.7

- **Step 2a:** The set of common vectors identified in step 2 is further filtered to remove the synonymous duplicates (e.g., $\overrightarrow{a}\,\overrightarrow{b}\,\overrightarrow{c}\,\overrightarrow{d_1}$, $\overrightarrow{a}\,\overrightarrow{b_1}\,\overrightarrow{c}\,\overrightarrow{d}$, $\overrightarrow{a}\,\overrightarrow{b_1}\,\overrightarrow{c}\,\overrightarrow{d_1}$) of a vector (e.g., $\overrightarrow{a}\,\overrightarrow{b}\,\overrightarrow{c}\,\overrightarrow{d}$), as follows. Using the BF bit indices from all the rows that contain the common vector ids a "common vector BF" is quickly constructed (Fig. 5.21)

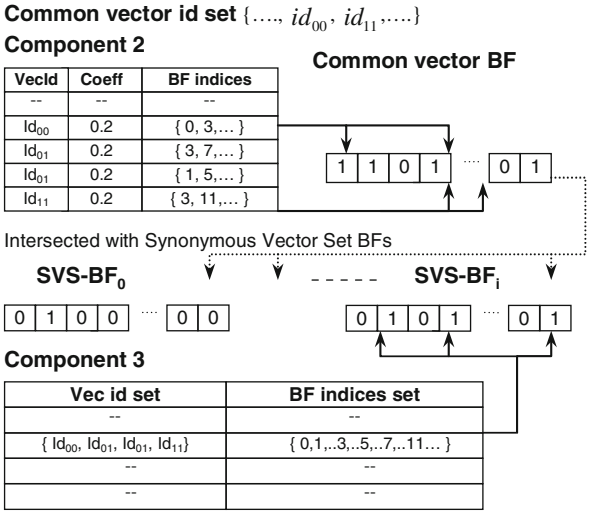


Fig. 5.21 Removing synonymous duplicates during descriptor comparison [9, 45]

Then we iterate (outermost loop in the nested iteration) over the "set of synonymous vector sets" and for each element, a synonymous vector BF is quickly constructed using BF indices and intersected with the common vector BF. If this intersected BF is not empty, we perform a nested iteration over this set of synonymous vector ids (in the middle loop) and over the set of common vector ids (in the innermost loop) to identify the ids that are in both sets. All subsequent common ids except the first one are dropped from common vector id set.

Extensions for hypernyms. To enable searching for a resource by a generic term (e.g., "Auto immune disease") instead of using the specific term for the concept (e.g., Type 1 diabetes), we have to incorporate all possible hypernyms for each term for similar reason as in case of synonyms. We consider a general model where the similarity value between a specific basic concept and its generic form is less than 1. The proposed hypernym incorporation technique is similar to that of synonyms. In the given example (Fig. 5.22) we replace $\vec{c}$ in $s_{abcd}\,\vec{a}\,\vec{b}\,\vec{c}\,\vec{d}$ by a sum of scalar-weighted hypernyms, $\vec{c_1}, \vec{c_2}, \ldots \vec{c_n}$, from the hypernym chain (Fig. 5.22) where $\vec{c}$ corresponds to the term in the bottommost node. This chain is obtained from "is a" taxonomies or ontologies (e.g., Disease Ontology [77]). The hypernymous polyads have ranks and can be ordered to form a "hypernym sequences": $\vec{a}\,\vec{b}\,\vec{c}\,\vec{d} > \vec{a}\,\vec{b}\,\vec{c_1}\,\vec{d} > \vec{a}\,\vec{b}\,\vec{c_2}\,\vec{d} > \ldots > \vec{a}\,\vec{b}\,\vec{c_n}\,\vec{d}$. A set of all such ordered sequences form the fourth component, the "set of hypernym sequences."

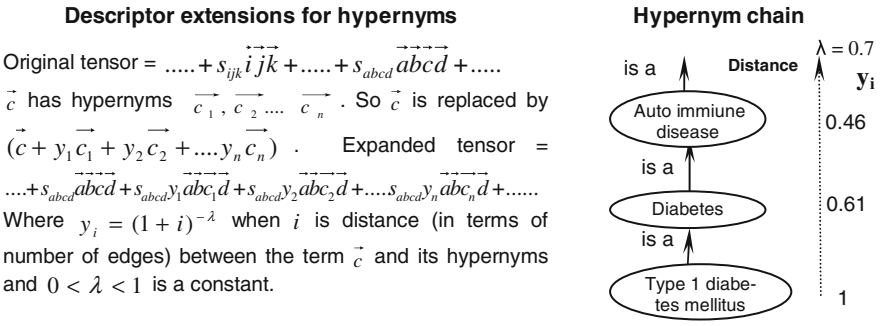


Fig. 5.22 Incorporation of hypernyms in descriptor generation and comparison algorithms [9, 45]

The data structure generating algorithm is extended by additional steps paired with the steps for the synonyms:

- **Steps 4a (paired with step 4) and 5a (with step 5):** The 128-bit vector ids for all the hypernymous vectors are generated and inserted in the BF.
- **Step 6a:** Each of the vector ids of a set of hypernymous vectors are inserted in additional rows in the coefficient lookup table. The coefficient columns of these additional rows are filled with the coefficient s_{abcd} multiplied with the respective terms y_m for each hypernymous vector $\vec{a}\,\vec{b}\,\vec{c_m}\,\vec{d}$ and with the set of BF bit indices.

- **Step 7a:** The set of hypernymous vector ids is grouped together as a set and paired with the set of BF bit indices to generate "set of hypernym sequences." We extend the inner product computation by a step 2b paired with step 2a:
- **Step 2b:** The set of common vectors identified in step 2a is further filtered to remove the hypernymous vector duplicates. This is achieved in a manner similar to step 2a with only two differences. Here instead of "set of synonymous vector sets" the "set of hypernym sequences" is used. While performing the nested iteration, when a second common vector id is found between the set of common vector ids and the current hypernym sequence, the common vector id having the lowest rank in the sequence is dropped to get the final filtered common set of vectors.

5.6 Discussions

The concept tree and its algebraic representation are not meant to replace natural language techniques, but to represent an entire complex concept using controlled terms for indexing purpose. However, these techniques can be used to solve problems beyond the primary purpose (indexing). A composition skewed toward conjunction and having the form as {<concept name>, <attributes>} defines a concept with more specificity compared to a single basis vector having form <concept name> (e.g., {"diabetes," "PTPN22"} can be used instead of "diabetes" alone to represent the object in Fig. 5.11). Similarly compositions like {"mouse," "animal"} or {"mouse," "computer device"} can be disambiguated to mean the rodent or the computer device. In this manner small compositions having small number of leaves can be used instead of a single basis vector to bolster the existing vector models [9] wherever such opportunity exists.

For scientific publications, the leaf terms are controlled scientific index terms, which convey complex concepts themselves. This enables smaller trees to convey more complex meanings compared to the vector representations. The number of leaves generally came to ~10 and number of levels ~4 or less. Such trees can be represented with moderate numbers of basis vectors. The memory footprint of these tensor-based keys remained manageable such that a large number of such keys can be accommodated in high-density memories (~GigaByte on single chip). As even higher density memories (~TeraByte on single chip) become available in future, the limit on tree sizes will get relaxed. However, there are ways (e.g., by using more advanced binder functions) to reduce the number of basis vectors without trading off the fidelity of the comparison accuracy. Adopting such methods improves the scalability on the proposed scheme [9].

If there is hash collision during vector id generation, the proposed dot product algorithm will produce wrong values. The likelihood (probability) of such an event is in the order of $p_{\text{collision}}{}^{*}(n_1{}^{*}n_2{+}n_1{}^{2}{+}n_2{}^{2})$, where "$p_{\text{collision}}$"= probability of hash collision and "n_1," "n_2" are the number of basis vectors in each tensor. For a high-quality 128-bit hash function, "$p_{\text{collision}}$" is sufficiently small $\sim 2^{-128}$ and for $n_1, n_2 < 10^4$ (a reasonably good budget), wrong computations are unlikely [9].

5.7 Conclusion

Searching is a key function in scientific cyber-infrastructure. The search technology should be able to search for objects from specialized scientific databases. Existing Internet search engines do not have this capability. In addition, the search technology should also provide more sophisticated meaning-based searching, compared to what is possible by existing meaning-based search engines. This calls for new kinds of semantic technologies that can enable sophisticated meaning-based searching. These required semantic technologies should enable computers to comprehend meaning of the objects being searched and user's search intentions, compare these meanings, and discern which object may satisfy user's need.

In addition, other kinds of semantic technologies are required to ease and facilitate the electronic publication process. It is desirable that the cyber-infrastructure allow users to describe meaning of objects being published using natural languages. The idea is that during publishing the human publisher will provide a natural language description of the object instead of just providing a few tag terms or a RDF description of the object as required by the semantic web paradigm [81]. The text objects will not need additional description as the text content itself will suffice. Users who want to search objects will enter their search intentions in natural language texts instead of just providing search terms.

To enable this kind of publishing and search operations, we need techniques to capture complex meaning from text narratives that describe objects and search intentions. We also need methods to compare the meanings conveyed by these texts in a manner that is coherent with human cognitive processes. This will require assimilating knowledge from cognitive sciences and linguistics to create computational models that represent and compare meanings within computers.

To materialize intention-based searching over large grids a distributed search technique is required as illustrated in [15, 17]. However, the operations of such distributed index or distributed searching on grids also require the same semantic technologies mentioned above.

In this chapter, as meaning representation and comparison is the key problem, we reviewed the basic concepts and existing work regarding meaning representation and comparison from cognitive science, linguistics, and computer science domains. With this given background, we presented a design of meaning representation and comparison technique which is coherent to the cognitive science and linguistics models. This proposed design addresses the key requirement of meaning compositionality which had not been addressed adequately and efficiently by existing works. We presented an algebraic theory and techniques to represent hierarchically composed concepts as a tensor which is amenable to efficient semantic similarity computation. Next we delineated a data structure for the semantic descriptors/keys and an algorithm to generate them and described an algorithm to compute the semantic similarity of two given descriptors (tensors). This meaning comparison technique discerns complex meaning while enabling search query relaxation [82] and search key interchangeability. This is achieved without the need of a meaning knowledgebase (ontology) unlike in [82]. The computation models presented here are suitable for efficient execution on simple hardware accelerators.

References

1. PubMed, http://www.ncbi.nlm.nih.gov/pubmed/. Accessed 1 Feb 2009
2. SRS Server at EMBI-EBI: http://srs.ebi.ac.uk. Accessed 1 Feb 2009
3. The SAO/NASA Astrophysics Data System: http://adswww.harvard.edu/. Accessed 1 Feb 2009
4. NCAR Community Data Portal (CDP) http://cdp.ucar.edu/. Accessed 1 Feb 2009
5. California Water CyberInfrastructure: http://bwc.lbl.gov/California/california.htm. Accessed 1 Feb 2009
6. CUAHSI Hydrologic Information System (CUAHSI-HIS) http://his.cuahsi.org/. Accessed 1 Feb 2009
7. Baker, K.S., Ribes, D., Millerand, F., Bowker, G.C.: Interoperability strategies for scientific cyberinfrastructure: Research and practice. Proceedings of the American Society for Information Science and Technology 42(1) (2005)
8. Bergman, M.K. White paper: The deep web: Surfacing hidden value. The Journal of Electronic Publishing 7(1) (2001)
9. Biswas, A., Mohan, S., Panigrahy, J., Tripathy, A., Mahapatra. R.: Enabling intention based search. Technical Report, Department of Computer Science, Texas A&M University (2008)
10. Medical Subject Headings, (MeSH): U.S. National Library of Medicine, www.nlm.nih.gov/mesh/. Accessed 1 Feb 2009
11. OCR: Optical Character Recognition http://www.cdac.in/html/gist/research-areas/ocr.asp. Accessed 1 Feb 2009
12. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing & Management 24(5) (1988) 513–523
13. Knecht, L.: PubMed: Truncation, Automatic Explosion, Mapping, and MeSH Headings, NLM Technical Bulletin 1998 May–June, 302 (1998)
14. Sowa, J.F. Semantic networks. In: Shapiro, S.C. (ed.) Encyclopedia of Artificial Intelligence, Wiley. http://www.jfsowa.com/pubs/semnet.htm (1992). Accessed 1 Feb 2009
15. Biswas, A., Mohan, S., Mahapatra, R.: Search co-ordination with semantic routed network. In: Proceedings of the 18th International Conference on Computer Communications and Networks, US Virgin Islands (2009)
16. Watts, D.J.: Six Degrees: The Science of A Connected Age. W.W. Norton & Company, New York (2003)
17. Biswas, A., Mohan, S., Mahapatra, R.: Optimization of semantic routing table. In: Proceedings of the 17th International Conference on Computer Communications and Networks, US Virgin Islands (2008)
18. Culicover, P.W., Jackendoff, R.: Simpler Syntax. Oxford linguistics, Oxford University Press, Oxford (2005)
19. Bai, C., Bornkessel-Schlesewsky, I., Wang, L., Hung, Y., Schlesewsky, M., Burkhardt, P.: Semantic composition engenders an N400: Evidence from Chinese compounds. NeuroReport 19(6) (2008) 695
20. Brennan, J., Pylkknen, L.: Semantic composition and inchoative coercion: An MEG study. In: Proceedings of 21st Annual CUNY Conference on Human Sentence Processing, University of North Carolina, Chapel Hill (2008)
21. Grodzinsky, Y. The neurology of syntax: Language use without Broca's area. Behavioral and Brain Sciences 23(1) (2001) 1–21
22. Piango, M.M.J.M.: The neural basis of semantic compositionality. In session hosted by the Yale Interdepartmental Neuroscience Program, Yale University (2006)
23. Murphy, G.: Comprehending complex concepts. Cognitive Science 12(4) (1988) 529–562
24. Culicover, P.W., Jackendoff, R.: The simpler syntax hypothesis. Trends in Cognitive Sciences 10(9) (2006) 413–418
25. Kirshner, H.S.: Language studies in the third millennium. Brain and Language 71(1) (January 2000) 124–128

26. Hagoort, P.: On Broca, brain, and binding: A new framework. Trends in Cognitive Sciences 9(9) (2005) 416–423
27. Caramazza, A., Berndt, R.S.: Semantic and syntactic processes in Aphasia: A review of the literature. Psychological Bulletin 85(4) (1978) 898–918
28. Kuperberg, G.: Neural mechanisms of language comprehension: Challenges to syntax. Brain Research 1146 (2007) 23–49
29. Friederici, A.D., Opitz, B., Cramon, D.Y.: Segregating semantic and syntactic aspects of processing in the human brain: An fMRI investigation of different word types cereb. Cortex 10 (2000) 698–705
30. Ye, Z., Zhou, X.: Involvement of cognitive control in sentence comprehension: Evidence from erps. Brain Research 1203 (2008) 103–115
31. Ekiert, M.: The bilingual brain. Working Papers in TESOL and Applied Linguistics 3(2) (2003)
32. Kim, K.H.S., Relkin, N.R., Hirsch, J.: Distinct cortical areas associated with native and second languages. Nature 338 (1997) 171–174
33. Pinango, M.: Understanding the architecture of language: The possible role of neurology. Trends in Cognitive Sciences 10(2) (2006) 49–51
34. Zurif, E.: Syntactic and semantic composition. Brain and Language 71(1) (2000) 261–263
35. Bickerton, D.: Language & Species, The University of Chicago Press, Chicago & London (1990)
36. Jackendoff, R.: Compounding in the parallel architecture and conceptual semantics. In: Lieber, R., Stekauer, P. (eds.) The Oxford Handbook of Compounding. Oxford Handbooks in Linguistics. Oxford University Press, Oxford (2009)
37. Collins, A.M., Loftus, E.F.: A spreading-activation theory of semantic processing. Psychological Review 82(6) (November 1975) 407–428
38. Anderson, J.R.: A spreading activation theory of memory. Journal of Verbal Learning and Verbal Behavior 22 (1983) 261–295
39. Anderson, J.R., Pirolli, P.L. Spread of activation. Journal of Experimental Psychology: Learning, Memory, & Cognition 10 (1984) 791–799
40. Saffran, E.: The Organization of semantic memory: In support of a distributed model. Brain and Language 71(1) (2000) 204–212
41. Rodriguez, R.A.: Aspects of cognitive linguistics and neurolinguistics: Conceptual structure and category-specific semantic deficits. Estudios Ingleses de la Universidad Complutense, 12 (2004) 43–62
42. Rajapske, R., Denham, M.: Fast access to concepts in concept lattices via bidirectioanl associative memory. Neural Computation 17 (2005) 2291–2300
43. Rajapske, R., Denham, M.: Text retrieval with more realistic concept matching and reinforcement learning. In Information Processing and Management 42 (2006) 1260–1275
44. Andersen, C.: A Computational model of complex concept composition. Master's thesis, Department of Computer Science, University of Texas at Austin (1996)
45. Biswas, A., Mohan, S., Panigrahy, J., Tripathy, A., Mahapatra, R.: Representation and comparison of complex concepts for semantic routed network, In: Proceedings of 10th International Conference on Distributed Computing and Networking (ICDCN). Hyderabad (2009)
46. Wolff, K.E.: A first course in formal concept analysis, F. Faulbaum StatSoft '93, 429–438, Gustav Fischer Verlag (2004)
47. Qi, J., Wei, L., Bai, Y.: Composition of concept lattices. Proceedings of the 7th International Conference on Machine Learning and Cybernetics, Kunming (July 2008)
48. Murphy, G.L., Medin, D.L.: The role of theories in conceptual coherence, in Psychological Review (1985)
49. Foggia, P., Sansone, C., Vento, M.: A performance comparison of five algorithms for graph isomorphism, 3rd IAPR TC-15 workshop on graph-based representations in Pattern Recognition (2001) 188–199

50. Ogata, H., Fujibuchi, W., Goto, S., Kanehisa, M.: A heuristic graph comparison algorithm and its application to detect functionally related enzyme clusters. Nucleic Acids Research 28(20) (2000) 4021–4028
51. Mitchell, J., Lapata, M.: Vector-based models of semantic composition. In: Proceedings of ACL-08: HLT, Association for Computational Linguistics, Columbus, Ohio (2008) 236–244
52. Widdows, D.: Semantic vector products: Some initial investigations. In: Quantum Interaction: Papers from the Second International Symposium, Oxford (2008)
53. Widdows, D. Geometric ordering of concepts, logical disjunction, and learning by induction. Compositional Connectionism in Cognitive Science, AAAI Fall Symposium Series, Washington, DC, October (2004) 22–24
54. Widdows, D. Orthogonal negation in vector spaces for modeling word meanings and document retrieval. In: Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL), http://acl.ldc.upenn.edu/acl2003/main/ps/Widdows.ps (2003). Accessed 1 Feb 2009
55. Lin, D.: An information-theoretic definition of similarity. In: Shavlik, J.W (ed.) Proceedings of the 15th International Conference on Machine Learning. Morgan Kaufmann Publishers, San Francisco, CA (1998) 296–304
56. Rodríguez, A.: Semantic Similarity Among Spatial Entity Classes Ph.D. thesis, Department of Spatial Information Science and Engineering University of Maine (2000)
57. Veksler, V., Govostes, R., Gray, W.: Defining the dimensions of the human semantic space. In: Proceedings of the 30th Annual Meeting of the Cognitive Science Society, Austin, TX (2008)
58. Lindsey, R., Stipicevic, M.V.V.: BLOSSOM: Best path length on a semantic self-organizing map. In: Proceedings of the 30th Annual Meeting of the Cognitive Science Society. Washington, DC (2008)
59. Cederberg, S., Widdows, D.: Using LSA and noun coordination information to improve the precision and recall of automatic hyponymy extraction. In: Proceedings of Conference on Natural Language Learning (CoNLL), Edmonton, Canada (2003) 111–118
60. Maguitman, A.G., Menczer, F., Roinestad, H., Vespignani, A.: Algorithmic detection of semantic similarity. In: Proceedings of the 14th International Conference on World Wide Web (WWW '05) ACM, New York, NY (2005) 107–116
61. Rada, R., Mili, H., Bicknell, E., Blettner, M.: Development and application of a metric on semantic nets. Systems, Man and Cybernetics, IEEE Transactions 19(1) (1989) 17–30
62. Jeh, G.: Simrank: A measure of structural-context similarity. In: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada (2002)
63. Yang, D., Powers, D.M.: Measuring semantic similarity in the taxonomy of WordNet. In: Proceedings of the 28th Australasian Conference on Computer Science, Newcastle, Australia 38 (2005)
64. Cilibrasi, R.L., Vitanyi, P.M.B.: The Google similarity distance. In: IEEE Transactions on Knowledge and Data Engineering 19(3) (2007) 370–383
65. Widdows, D.: Unsupervised methods for developing taxonomies by combining syntactic and statistical information. In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology – Vol 1, North American Chapter of the Association For Computational Linguistics. Association for Computational Linguistics. Morristown, NJ (2003) 197–204
66. Lemaire, B., Denhière, G.: Incremental construction of an associative network from a corpus. In: Proceedings of the 26th Annual Meeting of the Cognitive Science Society, Hillsdale, NJ (2004) 825–830
67. Dorow, B., Widdows, D., Ling, K., Eckmann, J.P., Sergi, D., Moses, E.: Using curvature and Markov clustering in graphs for lexical acquisition and word sense discrimination. In: Proceedings of the 2nd Workshop organized by the MEANING Project (MEANING 2005), Las Vegas, Nevada, USA February 3–4 (2005)

68. Borgida, A., Walsh, T., Hirsh, H.: Towards measuring similarity in description logics. In: Proceedings of the 2005 International Workshop on Description Logics (2005)
69. Hau, J., Lee, W., Darlington, J.: A semantic similarity measure for semantic web services. In: Web Service Semantics: Towards Dynamic Business Integration, Workshop at WWW, London, UK volume 5 (2005)
70. d'Amato, C., Fanizzi, N., Esposito, F.: A semantic similarity measure for expressive description logics. In: Proceedings of Convegno Italiano di Logica Computazionale (CILC05), Rome, Italy (2005)
71. Janowicz, K. Sim-DL: Towards a Semantic Similarity Measurement Theory for the Description Logic ALCNR in Geographic Information Retrieval On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops, Montpellier, France (2006) 1681–1692
72. Resource Description Framework (RDF): www.w3.org/RDF/. Accessed 1 Feb 2009
73. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using wikipediabased explicit semantic analysis. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI), Hyderabad, India (January 2007)
74. Widdows, D.: A mathematical model for context and word-meaning. Lecture Notes in Computer Science (2003) 369–382
75. Bottini, N. et al.: A functional variant of lymphoid tyrosine phosphatase is associated with type I diabetes. Nature Genetics 36 (2004) 337–338
76. Gene Ontology, http://www.geneontology.org/. Accessed 1 Feb 2009
77. Disease Ontology, http://diseaseontology.sourceforge.net/. Accessed 1 Feb 2009
78. Irgens, F.: Tensors. In: Continuum Mechanics. Springer, Berlin, Heidelberg (2008)
79. Broder, A., Mitzenmacher, M.: Network applications of Bloom filters: A survey. In Internet Mathematics 1(4) (2002) 485–509
80. Ripeanu, M., Iamnitchi, A.: Bloom Filters – Short Tutorial, Computer Science Department, University of Chicago. www.cs.uchicago.edu/∼matei/PAPERS/bf.doc (2001). Accessed 1 Feb 2009
81. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web, Scientific American Magazine. http://www.sciam.com/article.cfm?id=the-semantic-web&print=true. Retrieved on 26 March 2008 (2001). Accessed 1 Feb 2009
82. Tempich, C., Staab, S., Wranik, A.: Remindin': Semantic query routing in peer-to-peer networks based on social metaphors. In: Proceedings of the 13th International Conference on World Wide Web. WWW '04. (2004) 640–649

Chapter 6
BSIS: An Experiment in Automating Bioinformatics Tasks Through Intelligent Workflow Construction

Yu Pan, Enrico Pontelli, and Son Cao Tran

Abstract Existing bioinformatics tools are extremely difficult to integrate and inter-operate, because of the diversity of service deployment techniques and the representational heterogeneity of biological data they are interacting with. This chapter presents the *BioService Integration System (BSIS)*, a general framework which provides a graphic-based workflow language. The language enables the description of workflows composed of abstract semantic Web services and references to biologically relevant data. The workflows constructed in BSIS can be instantiated through automated planning techniques and automatically executed by adapting the *Web Service Integration Framework (WSIF)*. A prototype implementation of the system is presented to demonstrate the effectiveness and efficiency of the approach.

6.1 Introduction

A typical, modern biological analysis requires an extensive use of bioinformatics resources, in the form of biological databases, repositories, and computational tools. These databases and software tools can be either downloaded and installed on the scientist's machine or can be accessed remotely, through Web-based forms and Web portals.

While local databases and computational tools may provide better performance in general, the task of maintaining and synchronizing these resources can be overwhelming, given the high dynamics of data evolution in the field and the wide variety of tools that are continuously developed and enhanced. The use of Web-based applications and portals relieves the user from the burden of maintaining local installations – however, this solution makes the process of creating analysis pipelines significantly more cumbersome, forcing the user to switch between different Web sites, often requiring the manual shepherding of data between different

Y. Pan (✉)
Department of Computer Science, New Mexico State University, Las Cruces, NM, USA
e-mail: ypan@cs.nmsu.edu

H. Chen et al. (eds.), *Semantic e-Science*, Annals of Information Systems 11,
DOI 10.1007/978-1-4419-5908-9_6,

portals. Moreover, data produced by one software tool may not be directly acceptable by another tool – due to the widespread lack of standard data formats, especially in several legacy applications. While several data conversion tools are available, these are not comprehensive and often scientists have to resort on manual data manipulations – which are tedious and error prone. Finally, if the same computational procedure needs to be applied to large collections of data, the manual approach quickly becomes impractical, without the help of an automated programming environment.

This state of affairs has spawned a number of proposals, aimed at facilitating the design, development and deployment of *software workflows/pipelines*, thus simplifying the execution of bioinformatics analysis processes. A significant boost to the research in this field has come from the availability of novel software technologies, that provide the foundations for the high-level and scalable development of scientific workflows. In particular, advances in the areas of programming languages (e.g., the emphasis on the creation of infrastructures for domain-specific languages [9, 21]) and in the area of Web services and the Semantic Web [5, 36] have been greatly beneficial to these endeavors.

In this Chapter, we present an experimental framework, called *BioService Integration System (BSIS)*, for the design and execution of bioinformatics workflows. BSIS provides a graphical computational workbench for bioinformaticians. Similar to other concurrently developed frameworks, BSIS provides a graphical interface for the design of workflows. The novelties offered by BSIS can be summarized as follows:

- BSIS is *service oriented* and extensible;
- BSIS supports *semantic discovery* and *composition* of services;
- BSIS provides a graphical and user-friendly interface.

To the best of our knowledge, no other existing bioinformatics computational workbench concurrently provides all these features.

6.2 Related Work

The recent literature demonstrates an increased interest toward the development of automated and semi-automated workflow design environments for bioinformatics. In the next sections, we provide a brief overview of some of the relevant approaches in this area, classified in increasing level of abstraction.

6.2.1 Custom Scripts

At the lowest level of abstraction, we encounter methodologies of workflow constructions based on custom-made scripts. These scripts are commonly encoded

using traditional scripting languages, such as Perl, Python, or using makefile or Windows batch files notations [15, 37]. These scripts are typically static and they are often the result of tedious and years-long manual labor (e.g., [40]). The development of custom-made scripts requires the user to be familiar with at least one type of scripting language. The level of abstraction of scripting languages is very low and the resulting workflows (beyond trivial ones) are often hard to maintain and evolve. Such scripts can hardly be made general and, more importantly, bioinformaticians are often not trained to handle complex programming tasks and handle the low level of details required.

6.2.2 Domain-Specific Programming Environments

Concerted efforts have been proposed to reduce the level of details required in using traditional scripting languages to code workflows for biomedical analyses. A traditional approach pursued by several researchers is to enhance traditional scripting languages and programming languages with modules and packages specifically designed to solve bioinformatics problems. Examples of bioinformatics-oriented programming environments include BioPerl [43], BioJava [39], BioPython [6], and BioRuby [19]. Combined with their host languages, these modules allow the users to write sophisticated programs for their computational tasks, hiding the details required to access frequently used tools and data repositories. For example, BioPerl achieves this result by mapping distinct data formats to a common object model for data manipulation. Similarly to the case of programming scripts, this approach requires proficiency in both the programming language and the extension modules.

A different twist in this research direction comes from the Darwin [14] project. Darwin is a completely new programming language, developed to assist users in writing code related to bioinformatics tasks (with particular focus on evolutionary biology applications). Darwin uses a C-like syntax, and it provides a compiler to translate programs written in Darwin to C programs. Darwin provides data types for representing biologically relevant objects, like genes and chromosomes, and it offers a standard set of operators for manipulating these objects. Programming in Darwin might be easier, because of its domain-specific features. However, as with other approaches in this category, it requires users to be familiar with the concepts of traditional imperative programming; furthermore, the use of Darwin introduces a steep learning curve. Observe also that Darwin does not provide features for discovery of new modules and for extensibility.

6.2.3 Integrated Analysis Environments

An additional level of abstraction is provided by the several integrated analysis environments for biomedical research. These environments have been proposed to facilitate the development of analysis workflows that are specific to certain branches

of biomedical research. These environments usually compile a variety of computational models together and they allow the users to choose among those that are more appropriate for realizing a specific analysis task.

For example, PartiGene [38] is a software for EST Clustering analysis; a menu-driven environment allows the users to assemble a complete analysis pipeline, starting from raw EST sequences processing, followed by clustering, assembling, annotating, and reporting. Other examples of integrated environments include *POY (Phylogeny Reconstruction via Optimization of DNA and other Data)* [27], MEGA (Molecular Evolutionary Genetics Analysis) [22, 46], and Mesquite [27].

The use of integrated analysis environments is welcomed by many researchers, because of its simplicity and it has gained popularity. For example, MEGA is one of the most widely used environments in phylogenetic analysis, and there is an extensive literature of evolutionary analysis results produced using MEGA. However, integrated environments are not extensible and they provide only a fixed and limited range of analysis options. The workflows provided are static, mostly generated from fixed templates, and do not scale to the analysis of non-standard data sets.

6.2.4 Workflow Systems

The greatest level of abstraction and flexibility is provided by environments that allow the scientist to assemble high-level descriptions of workflows.

The Biology Workbench [31], developed by the San Diego Supercomputer Center, is an example of *Web-based workflow system*. It consists of a set of scripts that link the user's browser to a collection of information sources (databases) and application programs. Users can choose which tools to execute at each step, composing them into a workflow. The advantage of the Biology Workbench is that it hides the interface of each tool into a uniform Web-based form, thus relieving the users from the need to interact with concrete, low level, data formats. A limitation of the Biology Workbench is its restriction to linear workflows – i.e., all computations must be performed sequentially, under the control of the users' mouse clicks. In particular, full automation is not achieved. Additionally, the maintenance burden (e.g., inclusion of new data formats or tools) is placed on the shoulders of the Workbench developers.

An alternative class of workflow systems, *local workflow systems*, rely on the local development of workflows, without the need to rely on remote Web portals. Some relevant examples of local workflow system are Taverna [16] and Kepler [1]. Taverna is part of the ^{my}Grid [44] initiative, while Kepler is an open source project supported by the National Science Foundation and the Department of Energy. Both these environments provide graphical user interfaces, that allow the users to graphically construct workflows consisting of Web services. In both methods, the workflow construction requires the identification of the specific services to be applied at each step. While part of the dynamic composition of services is taken care by the underlying engine, the capabilities of automated discovery and composition of services in these systems are limited.

6.3 An Overview of the BioService Integration System

The *BioService Integration System (BSIS)* builds on the principles of the Semantic Web [5] and Web service technologies [36]. Figure 6.1 shows the overall architecture of the BioService Integration System, which consists of four major components: (1) the Web service infrastructure, (2) the workflow language, (3) the planner, and (4) the executor.

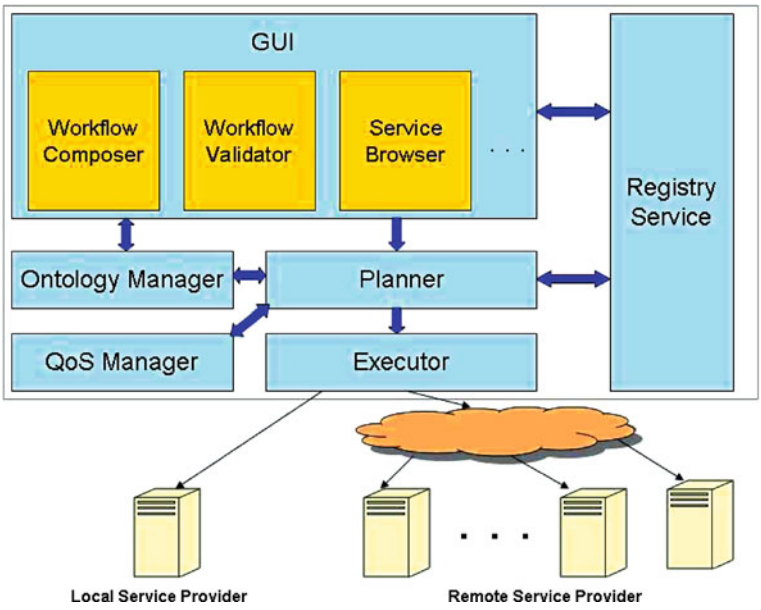


Fig. 6.1 The overall architecture of BSIS

6.3.1 The Web Service Infrastructure

The Web service infrastructure consists of service providers and service registries. By adopting the *Web Service Integration Framework* [11], service providers can be of any type – a HTML form-based service, a Java RMI service, a SOAP Web service, or even a local executable program. Their implementations are transparent to the end users and they are visible only through their descriptions built using the standard Web Services Description Language (WSDL) [7].

The service registries provide a location to store service information and they offer a set of interfaces for the end users to query the service information. In order to enable semantic discovery, we develop domain ontologies to annotate the services; the registry services provide the capability of querying the repositories using such semantic metadata.

6.3.2 The Workflow Language

BSIS defines a graphical workflow language and provides a graphical user interface for workflow composition and validation. Users of the system can construct workflows using the workflow editor and validate them against the syntactic rules of the language. The finished workflow can be *concrete* or *abstract*.[1] Intuitively, abstract workflows are under-specified workflows, containing operations that refer to *classes* of services instead of specific services. Abstract workflows must be *instantiated* through planning before execution of the workflow is possible.

6.3.3 The Planner

The planner considers an abstract workflow and instantiates it into a concrete and executable workflow. During this process, the planner communicates with registry services for service instance information, it consults the ontology manager for data-type information, and it relies on the quality-of-service manager to select among alternative service instances.

6.3.4 The Executor

If planning is successful, then the executor will proceed with the execution of the resulting workflow. The executor adopts the Web Service Invocation Framework and uses a factory pattern for the execution of different types of services, based solely on their Web services descriptions (in WSDL). The executor includes also an execution monitor.

In the following sections, each component of the system will be discussed in detail. In particular, Section 6.4 provides details of the Web service infrastructure; Section 6.5 describes the workflow design component of BSIS; Section 6.6 describes the planning infrastructure used by BSIS to make the workflow executable; Section 6.7 overviews the workflow execution infrastructure; finally, Sections 6.8 and 6.9 provide an evaluation of the infrastructure and discussion of possible optimizations and extensions.

6.4 The BSIS Web Service Infrastructure

The semantic Web services infrastructure is the basis for all other components in the system. It consists of *semantic Web services* – i.e., Web services whose descriptions have been annotated with semantic information drawn from domain ontologies – and *service registries*.

[1] This distinction will be explained in detail later in the chapter, when we discuss the workflow language.

Following the general view of Web services provided by the W3C, Web services are software systems designed to inter-operate in a networked environment. They are commonly realized by enhancing existing applications with Web APIs. In order to build *semantic* Web services, we adopt the Web Services Description Language (WSDL) [7] to attach semantic annotations to Web services. WSDL enables the discovery and automated composition of Web services. In the context of BSIS, we extend the concept of Web service to include any accessible application that can be described using an extended WSDL model.

6.4.1 Domain Ontologies

All metadata information used in the semantic description of Web services is defined in terms of concepts drawn from *domain ontologies*. Currently, two prototype ontologies have been introduced to characterize the bioinformatics domain: one for describing processes and transformations implemented by bioinformatics services – referred to as the *Service Ontology* – and the other for describing biological data – the *Data Ontology*. Observe that the BSIS system has been realized in such a way to reduce the dependency on any specific domain ontology. Users can configure the system to select any ontologies for annotating the components of their workflows.

The ontologies proposed in BSIS have been encoded using the standard Web Ontology Language (OWL) [4] and they are briefly discussed next.

6.4.1.1 Service Ontology

This ontology is used to define the different classes of typical bioinformatics applications and their mutual relationships. Figure 6.2 shows a hierarchical view of a segment of the service ontology (the complete ontology is discussed in [35]). At the highest level, the ontology organizes services in some general classes representing the most common transformations (e.g., alignment, phylogenetic analysis), while other classes provide description of non-transformation services, organized as (a) database retrieval, (b) storage and persistency, (c) visualization, and (d) comparison.

Service types are defined in the ontology as OWL classes along with properties that are applicable to them. For example, the following code fragment defines a service type class named Alignment_Multiple which is a subclass of the type `Sequence_Alignment`:

```
<owl:Class rdf:ID="Alignment_Multiple">
   <rdfs:subClassOf
         rdf:resource="#Sequence_Alignment"/>
</owl:Class>
```

The `Alignment_Multiple`, in turn, has subclasses that distinguish the alignment process based on the type of procedure, for example,

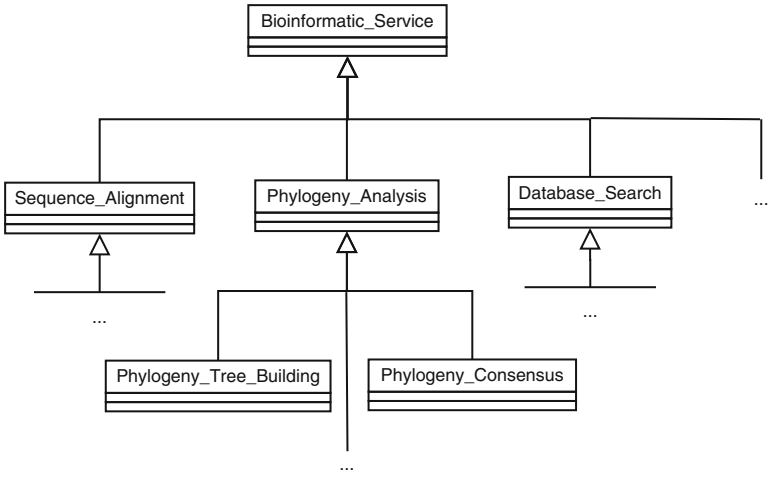


Fig. 6.2 Sketch of the service ontology

- `Maximum_Likelihood_Multiple_Alignment`,
- `Parsimony_Multiple_Alignment`, and
- `Distance_Based_Multiple_Alignment`.

The classes and their properties defined in the service ontology will be used to annotate Web services.

An OWL parser can parse this ontology and store each class definition in an *OWLClass* object, along with its properties. The object contains also all the relationships (e.g., descendents and ancestors) with other classes defined in the ontology. These pieces of information are crucial whenever we need to reason about the relationships between concepts defined in the ontology – e.g., during queries to find a service matching certain constraints.

6.4.1.2 Data Ontology

This ontology defines the types of data and data formats used in the most commonly used bioinformatics applications and their relationships. Figure 6.3 shows a hierarchical view of a segment of the data ontology. As in the case of the service ontology, the different classes of data are defined in the ontology as OWL classes and properties that are applicable to each specific type of data.

The root class of the data type ontology is `BioData`, which is a superclass of all other classes defined in the ontology. The direct children of `BioData` are `Number`, `String`, `Boolean`, `BioObject`, and `List`. The `List` class is the superclass of all classes of data that encode homogeneous sequences of data, while the `BioObject` class is a superclass of all data types of single structure (e.g., `Sequence`, `Nucleotide`, and `Tree`). Bioinformatics data are organized

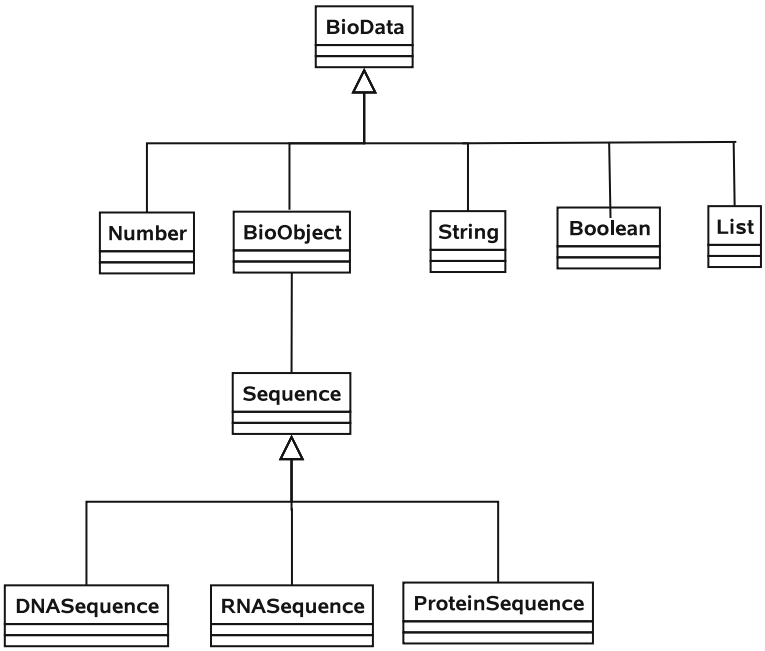


Fig. 6.3 Data ontology

in a style analogous to the organization adopted in the data ontology adopted in the myGrid project. The first level of subclasses of `BioObject` includes the following classes of entities:

- `Record`, e.g., describing protein family records (e.g., Pfam), protein structure record (e.g., SCOP record), etc.
- `Report`, describing a report produced by typical applications (e.g., BLAST report);
- `Location`, identifying data based on a location system (e.g., cellular location);
- `Sequence`, identifying different types of sequences (e.g., DNA, RNA);
- `Structure`, identifying a biological structure (e.g., RNA, protein).

Each of these classes are organized as sub-hierarchies.

As mentioned earlier, the system's behavior is parameterized by the ontologies chosen for describing data and services. These can be modified through a configuration file used by BSIS (e.g., see Fig. 6.4).

6.4.2 Services Description

Semantic Web services are Web services described with semantic information from a specific application domain – in our case, the bioinformatics domain.

```
<ontologies>
  <serviceOntology>
    <ontologyName>BSIS Service Ontology</ontologyName>
    <ontologyURL>http://www.cs.nmsu.edu/~ypan/ontologies/
                          bsis_services.owl</ontologyURL>
  </serviceOntology>
  <datatypeOntology>
    <ontologyName>BSIS Datatype Ontology</ontologyName>
    <ontologyURL>http://www.cs.nmsu.edu/~ypan/ontologies/
                           bsis_datatype.owl</ontologyURL>
  </datatypeOntology>
</ontologies>
```

Fig. 6.4 Configuration of ontologies

The functionalities of traditional Web services are described using WSDL Eric-01. WSDL uses XML schemas to describe the interface to the Web service. In particular, the description of the entities exchanged with the service are described focusing on the "syntactic" features of the service. Thus, the description is mostly focused on the operational aspects of the service and not on the semantic classification of data and operations.

Several approaches have provided solutions that enable the introduction of semantics in the description of Web services (e.g., [2, 30, 45]). Among them the DAML-S/OWL-S [2] approach is the most widely adopted method for semantic markup and composition of services. OWL-S defines an ontology that provides concepts and relations that can be used for describing Web services. The ontology consists of three sub-ontologies: service profile, service grounding, and process model. The service profile defines the functional and non-functional properties of the services. Service grounding contains information about service invocation. In an effort to align with WSDL, service grounding provides mappings from OWL-S atomic processes to WSDL operations. The process model describes how to define atomic processes and use them to form composite processes through workflow construction.

Although OWL-S is gaining acceptance in the Semantic Web arena, there are several reasons why we decided not to adopt it in BSIS:

1. OWL-S is significantly more complex than what required to support BSIS; indeed, the majority of the existing service repositories adopt syntactic (WSDL-based) service representations and do not refer to OWL-S.
2. At the time the BSIS project was launched, there were no established registry services that supported the OWL-S model. There have been some recent proposals in this area (e.g., [26]) for enhancing the UDDI model with OWL-S support, but these proposals are not mature.
3. The main focus of OWL-S is on its process model, which provides an ontology for describing workflow of services. Since we are creating our own workflow model, this feature is not significant to our needs. Furthermore, we found the OWL-S process model inadequate to meet the needs of describing BSIS

workflows – in particular, the lack of a notion of *variable* makes it cumbersome to relate inputs and outputs of different services used within a workflow.

4. The OWL-S model requires several non-standard extensions (through the addition of the *owl-s-parameter* attribute to the *wsdl:part* component and the addition of the *owl-s-process* attribute to the *wsdl:operation* component) to the WSDL description of a Web service. These extensions are not recognized by the Grimoires registry, which is the UDDI extended model that supports semantic discovery of services adopted in BSIS.

Indeed, several of these shortcomings have been recognized by other practitioners in the field (e.g., [3]).

In the context of BSIS, we propose instead a lightweight model for associating semantic metadata annotations to different components *inside* a WSDL description.

Figure 6.5 shows an example of a WSDL document for the description of the Clustal-W service [47] deployed at New Mexico State University. For simplicity, only relevant components of the WSDL document are shown. The components in lines 7–15 include the traditional WSDL description of the service – e.g., the type

```
1: <wsdl:definitions name="ClustalWService"
2:    targetNamespace="http://www.cs.nmsu.edu/bsis/services/clustalw"
3:    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
4:    xmlns:meta="http://www.cs.nmsu.edu/bsis/schema/metadata"
5:    ...>
6:    ...
7:    <wsdl:message name="ClustalWRequest">
8:        <wsdl:part name="sequences" element="types:sequences"/>
9:        ...
10:    </wsdl:message>
11:    ...
12:    <wsdl:portType>
13:      <wsdl:operation name="ClustalW">...</wsdl:operation>
14:    </wsdl:portType>
15:
16:    <meta:annotation>
17:      <meta:metadata ref="tns:ClustalW">
18:        <meta:metadataName>&rdf;type</meta:metadataName>
19:        <meta:metadataValue>
20:          <meta:value>&serviceType;Alignment_Multiple</meta:value>
21:        <meta:metadataValue>
22:      </meta:metadata>
23:      <meta:metadata ref="tns:sequences">
24:        <meta:metadataName>&rdf;type</meta:metadataName>
25:        <meta:metadataValue>
26:          <meta:value>&datatype;ListOfSequence</meta:value>
27:        <meta:metadataValue>
28:      </meta:metadata>
29:    </meta:annotation>
</wsdl:definitions>
```

Fig. 6.5 Annotated clustal-W service

of messages that can be accepted (message elements, lines 7–10) and the abstract operations provided by the service (portType elements, lines 12–14).

The *meta:annotation* element (lines 16–29) is an extension of the standard WSDL model, containing one or more metadata descriptions for some of the components (operations and message parts) in the WSDL description. Each metadata contains a name and a list of values, representing a collection of semantic information about the component identified through the *ref* attribute. For example, in Fig. 6.5, the metadata attached to *Clustal-W* indicates that the *type* of the service is *Alignment_Multiple* (line 20). The second metadata is attached to the WSDL message component named *sequences* and it states that the type of this message part is *ListOfSequence* (line 26).

6.4.3 Services Registry

Traditional Web service architectures make use of *UDDI (Universal Description, Discovery and Integration)* registries uddi for storing information about services. UDDI is limited in the support for semantic discovery of services based on metadata information. UDDI also provides relatively limited support for the execution of queries that target WSDL documents.

The *Grimoires* registry [25, 32] is an extension of the UDDI model, that provides support for querying WSDL documents and for the resolution of queries using metadata annotation. The Grimoires registry uses RDF models to store all metadata information. The Grimoires registry provides a number of functionalities that match our needs for implementing a bioinformatics service registry. In particular

- *Addition of metadata to entities*: this operation is used to associate a metadata component to an entity, where an entity is a named element within a WSDL document, e.g., a message part or an operation.
- *Metadata retrieval*: this operation retrieves all metadata attached to an entity.
- *Metadata-based querying*: this operation searches for entities containing specific components of metadata.
- *Operation-based querying*: this operation searches a WSDL interface according to one of its operations.
- *Message-based querying*: this operation searches an operation according to one of its message components.

As with other standard UDDI interface functions, these functionalities are exposed as Web services and invoked through SOAP messages. The server processes the requests, transferring them to RDF queries and using a RDF query engine. The results of a query are wrapped into SOAP messages as sent back as responses. When combined with the extended WSDL model, Grimoires provides us capabilities to perform semantic-based discovery of service providers, using metadata from domain ontologies.

6.5 Workflow Language

The BSIS workflow language is a graphic-based language, used to describe both *concrete* as well as *abstract workflows*, i.e., higher level workflows, that may not be immediately executable. A workflow is described by a directed graph structure. The workflow language introduces two types of entities to construct such graphs: *entity nodes* and *connectors*. Nodes and connectors are described by a number of properties.

In this section, we will first introduce the representation of nodes and connectors, followed by examples of different control structures; we will successively provide a more formal definition of the language.

6.5.1 Entity Nodes

The language provides four classes of entity nodes: *service* nodes, *data* nodes, *operator* nodes, and *control* nodes. These are summarized in Fig. 6.6. Purple (i.e., dark gray) nodes represent services, yellow (i.e., very light gray) nodes represent built-in operators, while light blue (i.e., medium gray) nodes represent data instances.

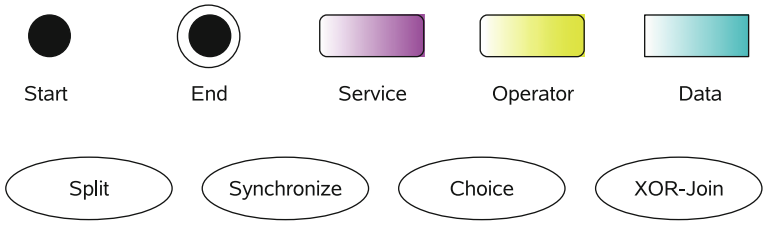


Fig. 6.6 Entity nodes

6.5.1.1 Service nodes

A *service node* contains information about a Web service. The service node can be either abstract or concrete. A *concrete* service node must have the following properties specified:

- A URL to the WSDL document containing the service description;
- The name of a portType within the WSDL document,
- The namespace of that portType, and
- The name of an operation within the port type.

These items uniquely identify an operation within a WSDL document, which we call a *service instance*. For example, a service instance of the *Alignment_Service*, with sample property values, is shown in Fig. 6.7. Observe that the designer of the

```
WSDL :                 http://www.cs.nmsu.edu/~ypan/wsdls/clustalw.wsdl
PortType Namespace :   http://www.cs.nmsu.edu/~ypan/services/clustalw
PortType :             ClustalWPortType
Operation :            Run
```

Fig. 6.7 Service instance properties

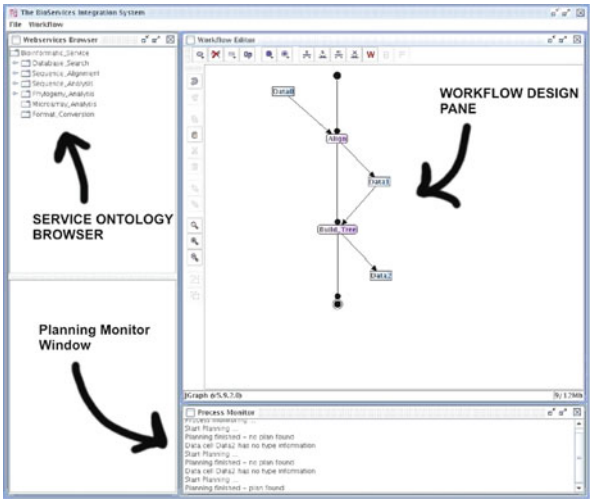


Fig. 6.8 Graphical interface for the creation of a workflow

workflow does not need to manually provide these properties – they are automatically provided whenever the user drags and drops a service into the workflow from a menu of services (see Fig. 6.8).

Compared to a concrete service node, an *abstract* service node need not specify any of these properties. Rather, an abstract service must have a *type information*, that identifies the service category – defined in terms of the service ontology. Moreover, the user can specify additional metadata information for the abstract service, that will be used in searching for a matching concrete service during the planning process. Each metadata represents a user constraint, that may be satisfied by the matching concrete service instance. For example, we could specify a metadata constraint to the *Tree_Building* service as follows:

Metadata name	Metadata value	Is optional?
`&rdf;type`	`&servicetype;` `#Phylogeny_Tree_building`	No
`&servicetype;algorithm`	`&serviceMeta;#PARSIMONY`	Yes
`&servicetype;software`	`&serviceMeta;#PHYLIP`	Yes

This metadata constrains the service to be of type *Phylogeny_Tree_Building* and it is preferable to find a service instance that is from the *PHILIP* package and that uses the *PARSIMONY* method for tree construction.

If the workflow contains at least one abstract service, then the workflow will be referred to as an *abstract workflow* and it cannot be executed directly. At this stage, the planner component will be executed to map all abstract services to their service instances, based on the user constraints on the service descriptions. On the other hand, a *concrete workflow* contains all the necessary service and flow information to produce an execution (performed by the executor component, described in Section 6.7).

6.5.1.2 Data Nodes

A data node represents a data item to be dealt with during the workflow execution. A data item can be an input to a service, provided by the user, or it can represent an output generated by an executed service. The user has the option to specify the data either in their textual raw format (i.e., as a string of text) or as a URI to a file containing the corresponding data. If the data is a user provided input, the I/O module of the executor will load the data from the corresponding data file. If the data is generated as output by a service, the I/O module will save the result to the data file at the specified location.

As in the case of abstract services, we can associate metadata information to a data node, that can be used as constraints during the planning process. For example, an input data could have the following metadata:

Metadata name	Metadata value	Is optional?
`&rdf;type`	`&datatype;#ListOfSequence`	No
`&datatype;sequenceFormat`	`&datatypeMeta;#FASTA`	Yes

This metadata indicates to the planner that the input data to the service *Alignment_Service* is of type *ListOfSequence* and has *sequenceFormat FASTA*. When searching for appropriate service instances, these constraints will be taken into consideration.

6.5.1.3 Control Nodes

Control nodes are employed to describe the control flow for the workflow execution. These nodes play an analogous role as control statements in a traditional programming language. Currently, there are six types of control nodes: *Start*, *End*, *Split*, *Synchronize*, *Choice*, and *XOR-Join*. Their meanings are described next:

1. *Start node*: this node represents the start of the workflow. For the sake of simplicity, we assume the *Start* node to have no incoming connections. Observe also

that if the *Start* node is connected to a data node, then such node represents one of the inputs of the workflow.
2. *End node*: this node represents the termination of the workflow. It is expected to have no outgoing connections. A connection from a data node to the *End* node indicates an output of the workflow.
3. *Split node*: The *Split* node indicates the creation of multiple execution paths, to be performed concurrently. For example, in Fig. 6.9, the execution is divided into three concurrent paths. The *Split* node is optional in the workflow. By default, if a node has multiple outgoing connections, it indicates a split event and each path is executed independently. However, the explicit use of *Split* may help clarify the execution path in a complicated workflow.

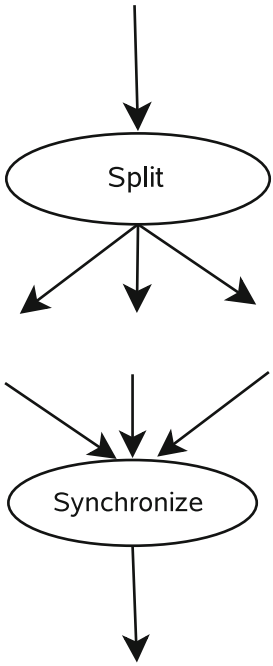


Fig. 6.9 Split and synchronize nodes

4. *Synchronize node*: The *Synchronize* node indicates the merging of multiple execution paths. It must have more than one incoming connection. For example, in Fig. 6.9 three execution paths are merged into one path. The execution cannot be continued until all three incoming paths finished execution.
5. *Choice node*: The *Choice* node divides the execution into multiple paths and it randomly selects one path for execution. The node must have more than one outgoing connection and must be used with the *XOR-Join* node to indicate the end of the divided paths. The divided paths must be independent of each other (i.e., there cannot be any connections between nodes on different paths), since only one of the path will be executed.

6. *XOR-Join node*: The *XOR-Join* node must be paired with a previous *Choice* node to indicate the end of multiple paths. It must have more than one incoming connection. The execution will be continued as long as one of the incoming path finished execution.

6.5.1.4 Operator Nodes

The operator nodes are used to represent simple local activities, such as string manipulations, set operations, file I/O, or mathematical computations. The system provides a variety of built-in operators for frequently used activities. It also allows users to define their own operators for specific application needs.

In order for the operators to participate in the planning process, they need to be annotated with semantic information, just as other service nodes in the workflow. However, there are no abstract operators. Every operator node must be bound to a specific *OperatorInfo* object at its creation time, which contains all the information about that operator. Figure 6.10 shows a sample annotation for the built-in *SequenceUnion* operator.

Fig. 6.10 Example of an operator annotation

```
<operator id="bsis.operator.SequenceUnion">
<description>
   Merges two lists of sequences
</description>
<inputs>
  <input name="SeqList1"
       type="&datatype;#ListOfSequence"/>
  <input name="SeqList2"
       type="&datatype;#ListOfSequence"/>
</inputs>
<outputs>
<output name="SeqList"
       type="&datatype;#ListOfSequence"/>
</outputs>
</operator>
```

6.5.2 Connectors

There are two types of connectors in the workflow language: *data flow* connectors and *control flow* connectors (see also Fig 6.11).

Fig. 6.11 Connectors

A *data flow connector* indicates the flow of information from one node to another node in the workflow. In general, data flow connectors can only exist between a data node and a service/operator node or between two service nodes or operator nodes. One complication arises when there is a data flow between two service/operator nodes which have multiple inputs/outputs. In this case, one must decide which output of one service is mapped to with which input of another service. The planner will give an educated guess first by using the data-binding algorithm. If the automated binding cannot resolve the ambiguity, the user will be asked to map the inputs/outputs explicitly.

A *control flow* connector indicates a dependence between nodes in the workflow. In other words, control flow connectors create a partial order among the nodes in a workflow, that the executor must use to plan the correct order of execution of the services in the workflow. Control flow cannot involve data nodes. All other nodes must be connected through control flow connectors or hidden control flow connectors. A hidden control flow connector is implicitly expressed through a data flow connector between a service and an operator or two services/operators. In this case, the data flow connector itself imposes an order constraint between the connected nodes.

6.5.3 Development of Sample Workflows

The language components described so far allow us to graphically construct workflows, with control structures not dissimilar to those encountered in traditional programming languages, such as sequencing, conditional executions, loops, concurrency, and selective executions. Figure 6.12 shows how these control flow structures can be expressed using these workflow components.

In Fig. 6.12, the *Sequence* workflow shows a sequential execution of five services for a typical phylogenetic study. The workflow takes a sequence as input, performs a *Blast* search against some database, analyzes the *Blast* report to extract a list of sequence identifiers, invokes a database search service to retrieve the actual sequence data, performs a multiple sequence alignment, and finally builds a phylogenetic tree.

The second workflow illustrates how conditionals and loops are constructed. The workflow starts from a multiple sequence alignment, followed by some alignment quality measurement service. The resulting score is compared with a predefined threshold; if the score is larger, then the *GreaterThan* operator will return *true*, the *FailIfFalse* operator will pass through, and the *BuildTree* service will be invoked on the alignment result. Otherwise, the *GreaterThan* operator will return *false* and the *FailIfTrue* operator will pass through. A *Refinement* operator will be invoked on the original data to improve the input quality and the execution will restart from the beginning. In this example, we can see that a conditional execution is achieved using the two special operators, *FailIfFalse* and *FailIfTrue*,

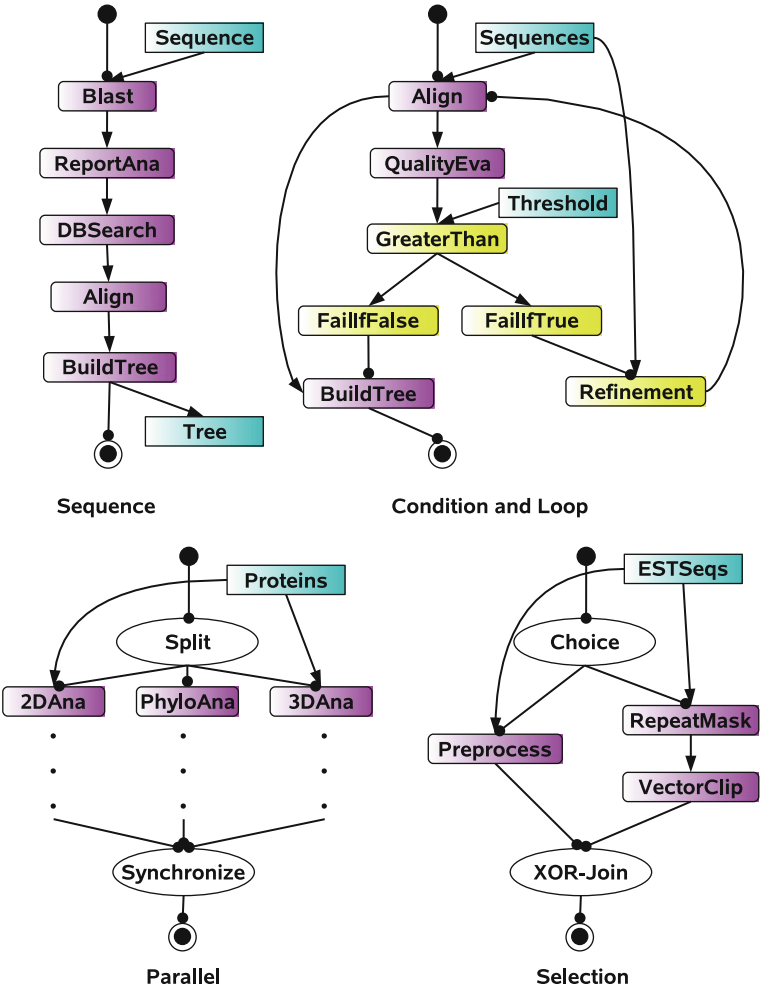


Fig. 6.12 Control flow structures

that take a Boolean value as input and fail (continue) if the input is false (true) (and vice versa for *FailIf True*). Loops are constructed when the workflow contains cycles.

The third example illustrates a simple parallel execution case, where a researcher might be interested in learning about secondary and tertiary structures of a protein, as well as performing a phylogenetic study using the protein sequence.

The last workflow illustrates a selective execution case, where a researcher might choose to pre-process a set of EST sequences using one service or invoke a *Repeat Mask* service first, followed by a *Vector Clipping* service. The choice of which branch to execute will be determined at run-time by the workflow executor.

6.5.4 Workflow Language Formalization

This section provides a short formal definition of a BSIS workflow and related concepts.

A *BSIS Workflow* is a graph (*V, E*):

1. V is a finite set of vertices $C \cup S \cup D \cup O$,
   a. C is a finite set of control nodes
   b. S is a finite set of service nodes
   c. D is a finite set of data nodes
   d. O is a finite set of operator nodes
2. E is a finite set of edges $CF \cup DF$
   a. CF is a finite set of control flow connectors
   b. DF is a finite set of data flow connectors

A *valid* BSIS Workflow is a BSIS Workflow conforming to the following syntactic constraints:

- $|Start| = 1$ and $|End| = 1$, i.e., it must contain one and only one *Start* and *End* nodes.
- $DF \subseteq (D \times (S \cup O)) \cup ((S \cup O) \times D) \cup (S \times S) \cup (O \times O) \cup (S \times O) \cup (O \times S) \cup (Start \times D) \cup (D \times End)$. A data flow can exist between data nodes and service/operator nodes or between two service/operator nodes. It neither can exist between a control node and another node, nor can it exist between two data nodes. There are two exceptions: the *Start* node can have outgoing data flow connectors to data nodes (indicating data inputs to the workflow), and the *End* node can have incoming data flow connectors from data nodes (indicating data outputs from the workflow).
- $CF \subseteq (V \times V) \setminus ((D \times V) \cup (V \times D))$, i.e., control flow connectors cannot exist between a data node and another node.
- For each node (except for a data node) in the workflow, there exists a path from *Start* to *End* that contains the node. This rule ensures that each executable node lies in a path from *Start* to *End*.

6.5.5 A Prototype Implementation of the Workflow Language

The prototype implementation of the BSIS Workflow language uses the JGraph[2] package, which is an open-source Java implementation of graph components. It builds on the JTree component from the Java Swing library, and it allows the association of user-defined objects to each component of the graph.

Figure 6.13 shows a snapshot of the graphic user interface for the workflow construction. There are three areas within the application window. The left pane shows the service hierarchy constructed from the service ontology file specified in the configuration file. The right middle pane contains the workflow editor, where a user can build the workflow by inserting the entity nodes using the toolbox icons and drawing edges between nodes. The bottom right pane serves as a message board showing information about the workflow when performing various activities.

The user can double-click a component in the workflow to open the Object Property Editor, which shows information about the underlying object associated with that component. The user can modify these properties like metadata annotation and I/O bindings. Each type of component has different properties and Fig. 6.14 shows the object editor for data nodes in the workflow.

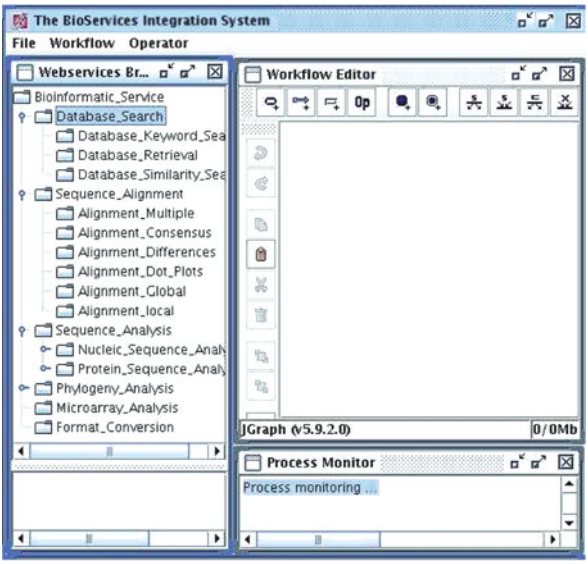


Fig. 6.13 Workflow construction tool

[2]Project home: http://www.jgraph.org.

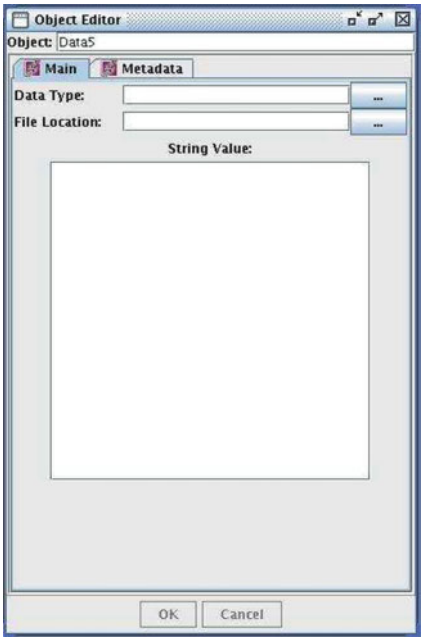


Fig. 6.14 Object editor

6.6 The Planner

6.6.1 Objectives of the Planner

A key feature of the BSIS workflow is the presence of abstract service nodes. As described in the previous section, a service node can be a *service instance*, in which case it requires a description composed of four pieces of information – i.e., a WSDL description, a portType name, a portType namespace, and an operation name – about the service instance, in order to be executable. Alternatively, a service node can be an *abstract service* node, with only metadata annotations specifying the user's requirements on the service provider.

For example, Fig. 6.15 shows an abstract service node with three metadata constraints: the *type* of the service must be *Phylogenetic_Tree_Building*, the algorithm used by the service must be *Parsimony*, and the *software package* from which the service is constructed must be *Phylip*.

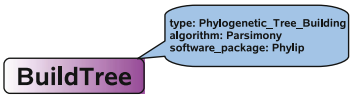


Fig. 6.15 An abstract service node

A workflow containing abstract service nodes is an abstract workflow which cannot be executed directly. The planner must be called to *instantiate* such workflow. During the planning process, the planner performs two tasks:

- *Service mapping.* This phase is used to identify, for each abstract service node, the matching service instances from the service registries that satisfy the user-specified metadata constraints.
- *Data binding.* After identifying a matching service instance, the planner must bind it to the service node in the workflow. This will determine the binding relationships among input/output parameters of the service instance and the data nodes/data flows in the workflow. Moreover, if the planner detects any data incompatibility, it will try to bridge the gaps by inserting appropriate services between the workflow components.

For the first task, queries to the service registry must be executed to identify the matching service instances. For the second task, the planner will make educated choices between I/O data bindings and involve the user in the process only if all efforts fail to resolve the ambiguity.

6.6.2 Service Mapping

When an abstract service node is created, it contains no more than a type information. This type information is a special metadata whose value is drawn from the service ontology. The use of abstract service nodes is helpful when the user of the workflow wants to perform certain operations, but he/she is not aware of any specific service providers. In this case, the user can just specify the type of that service and let the planner find out the matching instances from the service registry. Along with the type information, users can provide additional metadata, such as the algorithm used by the service or the software package from which the service should be chosen. All metadata information are extracted from the domain ontologies.

Once the type of a service is specified, the ontology parser will be able to extract just those properties that are applicable to that type of service. For example, Fig. 6.16 shows the Metadata Composer from the prototype implementation, that allows the user to add metadata to service nodes. The service node is of type `&servicetype;Alignment_Multiple`. The left pane of the Metadata Composer shows two metadata

```
&servicetypeMeta;multi_sequence_alignment_algorithm
 &servicetypeMeta;sequence_align_software_package
```

that are applicable to this service. The right pane shows the valid values for these metadata.

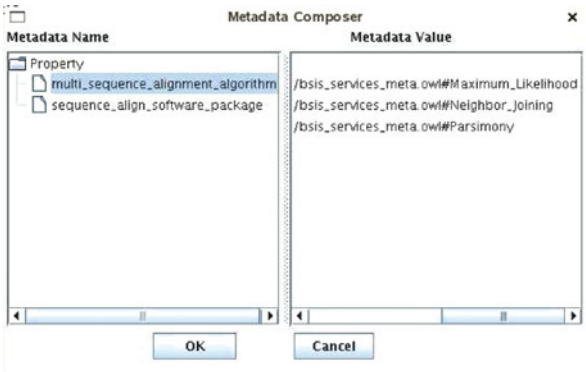


Fig. 6.16 Metadata composer

In order to provide greater flexibility, the metadata for a service can be specified as either mandatory or optional. A mandatory metadata constraint must be satisfied by the service instance – otherwise the service cannot be instantiated. Along with the metadata constraints imposed on the service itself, there are also constraints imposed by the input/output connections to the service. For example, Fig. 6.17 shows a situation where an abstract service has two incoming data flows and one outgoing data flow. In this case, *service0* has input constraints from *Data0* as well as input constraints imposed by the output of *service1*, if *service1* is instantiated before *service0*. It also has output constraints imposed by *Data2*. However, all these I/O constraints are considered optional constraints; in fact, if these are not satisfied, the planner will attempt to introduce additional steps (e.g., data conversions) to rectify the situation (as discussed later).

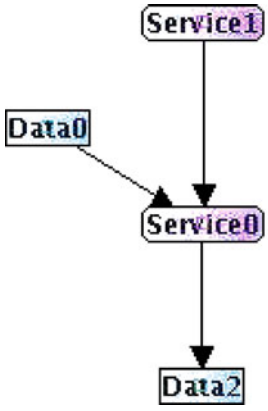


Fig. 6.17 A service with I/O constraints

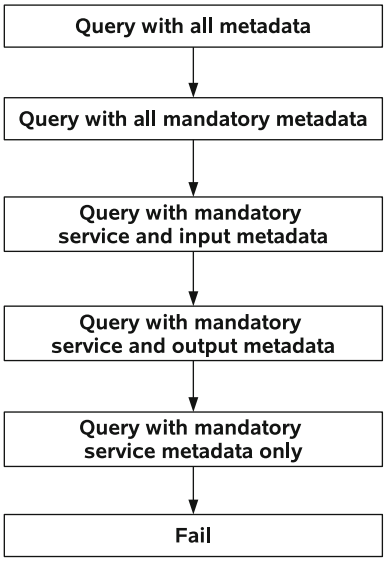


Fig. 6.18 Querying strategy

Ideally, all metadata constraints, whether they are on the service itself or on its inputs/outputs, can be wrapped into a single database query, and the registry service should be able to parse that information and return the satisfying results. However, current registry models are not flexible enough to support this querying strategy. For example, the registry service cannot differentiate a mandatory constraint from an optional constraint. As a result, multiple queries must be composed to obtain the service instance information satisfying the metadata constraints. In order to strike a balance between user expectations and query efficiency, the querying strategy of Fig. 6.18 and Algorithm 1 is used during the service-mapping process.

The method `ServiceMapping` tries to instantiate an abstract service using the following procedure. The first step is to obtain the current constraints on the service, which initially include all the constraints – i.e., the constraints on the service itself and on its inputs/outputs. A query is constructed to the registry using these constraints, and the resulting instances are returned, using the `QueryForInstances` method. An instance is selected from the results, using the method `PickAndRemoveInstance`, to bind to the service (the `UpdateConstraints` method). If the binding is successful, the `ServiceMapping` method successfully terminates. If the binding fails – various reasons could lead to a failure, discussed in the next sections – the inner loop will continue to select other instances for binding. If the instances are exhausted, the constraints will be relaxed (the `RelaxConstraints` method).

Algorithm 1 ServiceMapping

```
Require: Service
  repeat
    Instances = QueryForInstances(Service,GetConstraints(Service))
    success = False
    while success = False do
      if PickAndRemoveInstance(Service, Instances) then
        success = UpdateConstraints(Service)
      else
        break
      end if
    end while
    if success = True then
      break
    end if
  until not RelaxConstraints(Service)
```

Figure 6.18 shows the query relaxation strategy used during the service-mapping process. The planner starts by querying the registry with all metadata constraints. If this query fails to return any results, then only mandatory constraints will be used in the successive query. If this query fails as well, then only mandatory service and input constraints will be used for the next query. Finally, if the result returned by using only mandatory service constraints is empty, then the planner will fail. In other words, no service instance can be found to satisfy the minimum requirements imposed by the user. These constraints must be relaxed manually by the user before continuing.

6.6.3 Quality of Service

Generally, the planner will find a set of service instances for each abstract service node. The planner will have to select one service among them for binding. A natural choice is to select services based on their *quality*. There have been a variety of proposals for measuring service quality [28, 41, 49]. These approaches are either *global* – i.e., they combine measurements from different users to provide a unique score for a service provider – or *local* – they use only local preferences of a single user.

In our implementation, we use a simple method for comparing quality of services based on the history of service invocations by local users. We prefer local preferences over global preferences because of the following reasons:

- No global standard or well-established implementation techniques exist for measuring service quality.
- Different users may have different preferences for specific aspects of a service. Thus, it is very hard to combine different users' feedbacks in an objective way.
- Some quality aspects (for example, the network traffic and delay) may only be meaningful to a specific client environment.

For each service executed in BSIS, three pieces of information are saved for future use: number of failures, number of successes, and the most recent execution (time and whether it was a successful or a failed invocation).

Users of the workflow are given options to configure which of the following methods should be used to select among available services:

- `SELECT_MOST_RECENT_SUCCESS`. The service that succeeded most recently will be selected.
- `SELECT_SUCCESS_RATE`. The service that has the highest success rate will be selected.
- `SELECT_RANDOM`. A service is randomly selected.

In order to design for extensibility, BSIS uses a quality manager as a proxy for providing quality of service information. This manager provides an interface for registering new quality of service providers. Thus, if well-established quality standards become available, and an infrastructure for measuring and gathering such data is in place, it will be possible to write new service providers and register them with the system to replace the old service provider.

6.6.4 Data Binding

Once a service instance is selected for an abstract service node, it is necessary to make sure that the inputs and outputs of that service instance match the specification of the workflow, as far as data flow is concerned. For example, in Fig. 6.19, let us assume that *Service1* produces two outputs, while *Service2* accepts only one input.

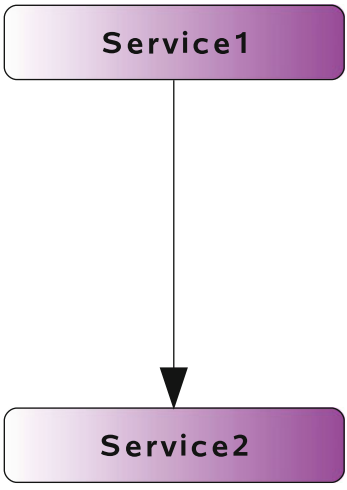


Fig. 6.19 Data binding between two services

In the first stage of data binding, we need to figure out which output of *Service1* flows into *Service2* using the metadata annotations.

Two principles are applied in sequence to make a binding decision. The *ancestor-descendent* principle checks to see if the data sender (the output of *Service1*) belongs to the same class or to one of the subclasses of the data receiver (the input of *Service2*), as far as their data classes are concerned. The classes and subclasses are described by the data ontology. If this is the case, then they can be bound together, but not vice versa. For example, if *Service1* in Fig. 6.19 produces a *ListOfAlignedSequences* and a *PhylogeneticTree*, while *Service2* accepts a *BioTree* and we know from the domain ontology that *PhylogeneticTree* is a subclass of *BioTree*, then these two data should be bound together.

When none of the outputs from *Service1* has an ancestor–descendent relationship with the input of *Service2* in the data ontology, the algorithm applies the *Closest-Common-Ancestor* principle to make an educated guess. This principle is based on determining the common ancestors for each output–input data pair in the data ontology. The output data that has the closest common ancestor with the input data will be bound together. For example, suppose that the types of the two outputs of *Service1* (O_1 and O_2) have the relationship with the type of the input of *Service2* (I_1) shown in Fig. 6.20 (according to the data ontology). In this case, I_1 will be bound to O_1, since they share a closer ancestor (A_1) compared to the ancestor of I_1 and O_2 (A_2).

When both principles fail to differentiate between alternative binding options (e.g., both outputs of *Service1* may have the same metadata annotations), then the user will be consulted to make the binding decision.

Several heuristics have been introduced to handle the other possible bindings cases, e.g., many-to-one bindings, many-to-many bindings, and to deal with binding between multiple services (e.g., one service to many services). These are mechanical generalizations – and they are discussed in details in [35].

So far, we have only discussed the data bindings between service nodes. Data bindings between other types of nodes (operator nodes and data nodes) can be fit into the above binding patterns. In this regard, an operator node behaves exactly as a service node and a data node can be seen as a service node with only one

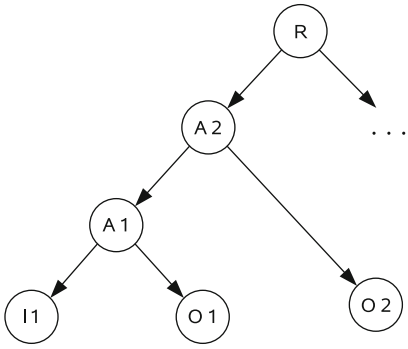


Fig. 6.20 Data-binding example

input/output. Furthermore, a data node to/from a service/operator node will always be bound before other types of bindings (service to service, operator to service, or service to operator) take place.

Often the binding parties (data sender/receiver) are not exactly the same and in many cases extra services are required to convert one data into another. For example, the output of *Service1* and the input of *Service2* may be both of type *ProteinSequence*, but they differ from each other when the former is in *GenBank* format and the latter is in *FASTA* format. Sometimes even their data types can be different. In these cases, data incompatibility events are triggered. In this case, the planner will initiate a new search to determine additional service(s) to perform format conversion before the workflow can be executed. This process is discussed next.

6.6.5 Data Conversion

Data conversion is part of the data-binding process. It occurs whenever a data incompatibility event is triggered. The objective of a data conversion is to expand the workflow through the insertion of additional service nodes. The service nodes will provide data transformations aimed at establishing compatibility between incompatible data.

The compatibility between data depends on their metadata annotations (observe that datatype is a special metadata). It is also important, when talking about compatibility, to differentiate two data items depending on their respective roles in the data flow. A data flow is always directional – thus, one data item will be the *receiver* while the other will be the *sender*.

Formally, a *sender (S)* is compatible with a *receiver (R)* if

1. *S* has exactly the same metadata as *R* – i.e., for each metadata of *S*, one can find a metadata of *R* that has the same name and values and vice versa; or
2. *S* is a *subclass* of *R*. The notion of subclass is defined as follows:

    a. For each metadata of *R*, one can find a metadata of *S* that has the same name. We will refer to these metadata as *name-equivalent* metadata.
    b. For each metadata (*MR*) of *R* and its name-equivalent metadata (*MS*)of *S*, there exist a metadata value (*VS*) of *MS* and a metadata value(*VR*) of *MR* such that *VS* is a subclass of *VR* in the considered domainontology. Here we consider subclass relationship to be reflexive, thus a class C is a subclass of itself.

By checking the properties of the data sender and the data receiver, the planner conducts either a *guided search* or a *blind search* for a data conversion. These are discussed next.

6.6.5.1 Guided Search

Under the *guided search*, the users of BSIS may define a set of rules that should be followed by the planner to find appropriate services for specific data incompatibility events. For example, Fig. 6.21 illustrates a search strategy defined for data with different sequence formats.

```
<searchStrategy meta="&datatypeMeta;sequenceFormat">
  <senderValue/>
  <receiverValue/>
  <sequence>
    <service>
      <metadata>
        <metadataName>&rdf;type</metadataName>
        <metadataValue>
          <value>&servicetype;Format_Conversion</value>
        </metadataValue>
      </metadata>
    </service>
  </sequence>
</searchStrategy>
```

Fig. 6.21 Search strategy example

The search strategy defined in Fig. 6.21 instructs the planner to insert a *Format_Conversion* service between the data sender and the data receiver when they differ in their *sequenceFormat*s. The *senderValue* and *receiverValue* sub-elements may also be specified for conversion between specific values. The *sequence* element identifies a series of service nodes that should be inserted to make such conversion.

There are no limits on the number of services that can be inserted during a guided search. However, only sequential composition of services is supported at this time.

6.6.5.2 Blind Search

If a search strategy has not been established for a data incompatibility event, a blind data conversion will be performed. In this case, the only information available is the identification of who is the data sender (the output data from previous service with its metadata constraints) and the data receiver (the input data of the current service with its metadata constraints). The goal is to find a sequence of services that can convert the data sender to the data receiver. The algorithm proceeds as follows:

1. The data sender is used as input constraint to query the registry for services that may take this data as input. This will return a set of matched service instances, referred to as IS_0.
2. The data receiver is used as output constraint to query the registry for services that may generate this data as output. This will return a set of matched service instances, referred to as OS_0.

3. The intersection of IS_0 and OS_0 is computed. If the resulting set is not empty, then one of the service instances in the resulting set is inserted in the workflow to perform the conversion. If the result set is empty, it means that no single service can realize such conversion and we will continue the search process as described below.
4. For each service instance in IS_0, we determine its output data. We create a data set combining all output data from all services in IS_0 (denoted by ID_1). For each data in ID_1, we repeat step 1 above and combine the results to get a new set of service instances (denoted by IS_1). We finally merge IS_0 and IS_1 to get IS'_1 .
5. For each service instance in OS_0, we determine its input data. We create a data set combining all input data from all services in OS_0 (denoted by OD_1). For each data item in OD_1, we repeat step 2 above and combine the results to obtain a new set of service instances, denoted by OS_1. Finally, OS_0 and OS_1 are merged to produce OS'_1 .
6. The intersection of IS'_1 and OS'_1 is computed. If the result set is not empty, then we can backtrack to find the sequence of services that can be inserted into the workflow to perform the conversion (we record the links between data and services at each expansion step). Otherwise we need to return to step 4 above and continue the search process.
7. The search process will stop when one of the following situations occurs:

    a. A solution is found. In this case, the intersection of IS'_n and OS'_n at some step n is not empty and we can extract the solution from the data-service links maintained at each expansion step.
    b. The search space is exhausted. In this case, neither IS'_n nor OS'_n are expanding, i.e., no new service instances are added to the set. This indicates that no solution can be found to perform the data conversion and the planning will fail.
    c. As an optimization strategy, the user may specify the maximum length of a plan that can be generated in the configuration file. When that number is reached and no plan is found, the planner will fail.

Figure 6.22 provides a graphical representation of the search algorithm described above. Notice that this is a bi-directional breadth-first search algorithm over the search space. The planner will find a shortest plan between the data sender and the data receiver, if such plan exists. A formalization of this search is presented in Algorithm 2.

Blind search should always be the last resort for data conversion. The reasons are that (1) it is usually time consuming and (2) it may result in plans that are not relevant to the computational tasks. The users of BSIS are encouraged to define search strategies for their specific problems or set up a max search depth in the configuration file. Libraries of search strategies are provided by BSIS to facilitate the task of the user.

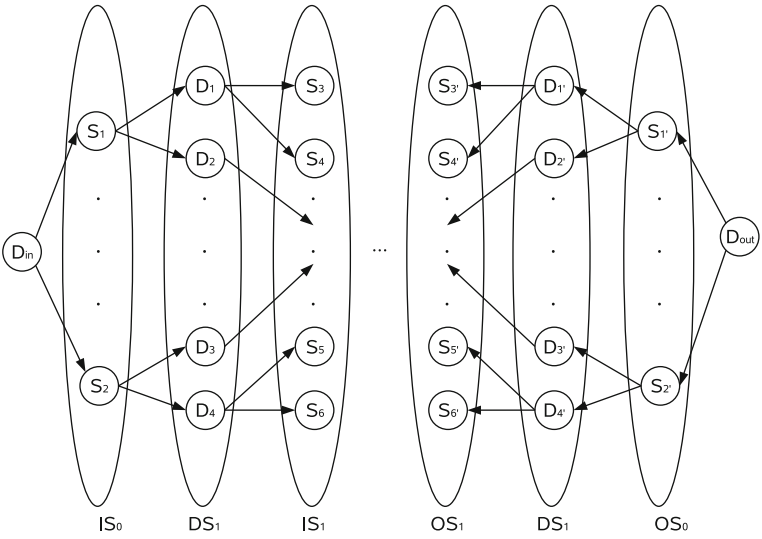


Fig. 6.22 Planning algorithm for data conversion

6.6.6 Planning the Workflow

So far, we have discussed how individual abstract service nodes are instantiated through the data-binding process. A BSIS workflow usually contains more than one abstract service node. The order in which the nodes are instantiated will greatly affect the performance of the planner, since they may return totally different workflows, in terms of the binding instances and the inserted data conversion plans.

The heuristics we use for determining service binding order is the *Most-Constrained-Service-First (MCSF)* heuristics. Abstract service nodes in the workflow are ordered by the number of user-specified constraints. The service node with the largest number of constraints is selected for binding. The rationale behind this heuristics is that the more constraints an abstract service has, the less service instances it will be mapped to. Since new mapped service instances will result in new constraints on existing service nodes, thus reducing the number of service instances that can be mapped to those service nodes, we will expect to find the solution faster or fail faster if the solution does not exist. If the binding process succeeds, then the workflow will be updated with new information from the binding instance (it may impose extra constraints to the service nodes in the workflow). Otherwise the planner backtracks to find alternative solutions.

6.6.7 Planning with an External Planner

We have explored an alternative architecture to support the process of constructing the executable workflow. The approach relies on the use of existing state-of-the-art planners to achieve the goal of instantiating an abstract workflow. The alternative approach builds on mapping the abstract workflow to a *Golog* [24] program with concurrent execution (or C-Golog), with an underlying situation calculus theory that will be used by the external planner to develop the concrete workflow. We will begin with a short introduction of the basic terminology of situation calculus and the language C-Golog.

Algorithm 2 Blind Search

Require: Sender
Require: Receiver
  IS_0 = QueryForInstances(Sender, GetInputConstraint(Sender))
  OS_0 = QueryForInstances(Receiver,GetOutputConstraint(Receiver))
  if $(IS_0 \cap OS_0 \neq \emptyset)$ **then**
    Select $S \in IS_0 \cap OS_0$
    Insert S in the workflow
    return True
  end if
  $i = 0$
  repeat
    $ID = \{d \mid S \in IS_i, d \in \text{GetOutputConstraint(S)}\}$
    $OD = \{d \mid S \in OS_i, d \in \text{GetInputConstraint(S)}\}$
    $IS_{i+1} = IS_i \cup \bigcup_{d \in ID} \text{QueryForInstances}(d, GetInputConstraints(d))$
    $OS_{i+1} = OS_i \bigcup_{d \in OD} \text{QueryForInstances}(d, GetOutputConstraints(d))$
    $i = i + 1$
    if $(IS_i \cap OS_i \neq \emptyset)$ **then**
      Select $S \in IS_i \cap OS_i$
      Insert path from Sender to Receiver through S in the workflow
      return True
    end if
  until $(IS_i = IS_{i-1}$ and $OS_i = OS_{i-1})$
  return False

6.6.7.1 Situation Calculus

Situation calculus is one of the most commonly used theoretical frameworks for the description of planning problems. In situation calculus, a dynamic domain is described in terms of the possible states of the domain (*situations*) and the actions that can modify the state.

The basic components of the situation calculus language, following the notation of [42], include

- a special constant S_0, denoting the initial situation,
- a binary function symbol *do*, where *do*(*a*,*s*) denotes the successor situation to *s* resulting from executing the action *a*,

- fluent relations of the form $f(s)$,[3] denoting the fact that the fluent f is true in the situation s, and
- a special predicate $Poss(a,s)$ denoting the fact that the action a is executable in the situation s.

A dynamic domain $\mathcal{D}$ can be represented by a theory containing

(i) axioms describing the initial situation S_0 – intuitively, a description of the initial configuration of the world;
(ii) action precondition axioms, one for each action a, characterizing $Poss(a,s)$ – intuitively, a description of what conditions should be satisfied to enable the execution of the action;
(iii) successor state axioms of the form $F(x, do(a, s)) \equiv \gamma^{+}_{F(x)}(a, s) \vee (F(x, s) \wedge \neg\gamma^{-}_{F(x)}(a, s))$ – one for each fluent F, stating under what condition $F(x, do(a, s))$ holds, as a function of what holds in s;
(iv) some additional foundational, domain-independent axioms.

The semantics of situation calculus allows us to prove/check whether a property φ (expressed as a propositional composition of fluents) holds after the execution of an action sequence $\alpha = [a_1, \ldots, a_n]$ starting from the initial situation; this is denoted by $\mathcal{D} \models \varphi(do(a_n, do(a_{n-1}, \ldots, do\,(a_1, S_0))))$ or $\mathcal{D} \models \varphi(Do([a_1, \ldots, a_n], S_0))$.

6.6.7.2 C-Golog

C-Golog is essentially the language Golog [24] extended with the concurrent execution construct of ConGolog [10]. The constructs of concurrent Golog are summarized in Fig. 6.23.

Fig. 6.23 Constructs of Golog

α	*primitive action*
$\phi?$	*wait for a condition*
$(\sigma_1; \sigma_2)$	*sequence*
$(\sigma_1 \| \sigma_2)$	*concurrent execution*
$(\sigma_1 \mid \sigma_2)$	*choice between actions*
$\pi x.\sigma$	*choice of arguments*
σ^*	*nondeterministic iteration*
if ϕ **then** σ_1 **else** σ_2	*synchronized conditional*
while ϕ **do** σ	*synchronized loop*
proc $\beta(\mathbf{x})\sigma$	*procedure definition*

Observe that the set of constructs of C-Golog is similar to the set of constructs in well-known programming languages such as C, JAVA. The basic blocks of a C-Golog program are actions and fluent formulae, whereas the basic blocks of a C/JAVA program are variables and assignment statements.

[3] A fluent is a proposition whose truth value can change over time.

The semantics of a C-Golog program can be described similarly to that of ConGolog [10]. The precise semantic definitions of Golog and ConGolog can be found in [24, 10]. Various extensions and implementations of Golog and ConGolog can be found at http://www.cs.toronto.edu/~cogrobo Web site.

6.6.7.3 Workflows as Golog Programs

We adopt the perspective of looking at Web services as actions in a situation calculus [29]. Roughly speaking, each Web service is viewed as an action with the following properties:

- The action name corresponds to the identification of the service;
- The arguments of the action correspond to the parameters of the service; and
- The effects of the action correspond to the outputs of the service.

More precisely, we will use $act(\mathbf{i}, \mathbf{o})$ to represent a Web service *act* with the list of input parameters **i** and the list of output parameters **o**. With a slight abuse of notation, we will also identify with $act(\mathbf{i}, \mathbf{o})$ a local activity *act* employed in an operator node with the list of input parameters **i** and the list of output parameters **o**. Observe that **i** may contain variables or constants. For example, the input parameters of a partially instantiated service might contain some constants. On the other hand, the input parameters of the service – in its full generality – contain only variables.

Systems have been proposed (see, e.g., [12, 20]) to translate Web services into actions, described using the Planning Domain Description Language (PDDL), which is the input language for several state-of-the-art planners. These systems can be easily adapted to produce a situation calculus theory instead, by (i) translating each action precondition into an executability condition of the action, and (ii) combining conditional effects of actions into successor state axioms for each fluents of the domain.

Let us discuss the steps required to create a C-Golog program P_W from an abstract workflow $W = (V, E)$ – where V are the nodes of the workflow and E are the connectors. We will restrict our attention to the case of workflows that are well structured – i.e., they correspond to structured uses of control constructs, where the control nodes are used to implement control structures analogous to those of traditional programming languages. Under this condition, a workflow can be easily converted into a C-Golog program through the following steps:

- Associate a unique integer to each node in W.
- Replace the label l of each service, operator, or data node, with l_{n_v}, where n_v is the integer associated to the node.
- Ignore all the data nodes and the links to/from the data nodes from the workflow.
- Translate the workflow into a set of programs, whose basic components are the labels of service and operator nodes of W, and the flows of the programs are specified by the control nodes of W. Moreover, we can create procedures to represent *Choice* and/or *Split* control nodes.

The programs obtained from the transformation of the workflows in Fig. 6.12 are as follows (using the constructs described in Fig. 6.23):

- **Sequence**:

```
( Blast1     ;
  ReportAns2 ;
  DiffSearch3 ;
  Align4     ;
  BuildTree5
)
```

- **Conditional and Loop**:

```
( Align1;
  QualityEva2;
  while Greater_Than3 do
   ( Refinement4;
     Align1;
   ) QualityEva2
  endwhile
) BuildTree5
```

- **Parallel**: $P_1||P_2||P_3$ where

P_1 : 2DAna$_1$;...;
P_2 : PhyloAna$_2$;...;
P_3 : 3DAna$_3$

- **Selection**: $P_1 \mid P_2$ where

P_1 : Preprocess$_1$;
P_2 : RepeatMask$_2$; VectorClip$_3$

Observe that the translation of the second workflow employs a standard method to convert a structured use of `goto` statements to a `while-loop`, and the translations of the third and fourth workflows induce some additional labels (e.g., P_1, P_2). The introduction of new labels is done for *Split* and *Choice* nodes, where each new label represents an outgoing link from these nodes.

The set of programs obtained from the above transformations is denoted by P_W. Observe that, by ignoring the data nodes, the workflow resembles a dataflow representation of a program in a parallel programming language with constructs `if-then-else`, `while`, `sequence`, `concurrent execution` (Split), and `nondeterministic choice` (Choice), which makes the last step in the translation possible.[4] It is easy to see that the program obtained from the above transformation is a C-Golog program if its components are basic actions and/or procedures encoding C-Golog programs.

In the rest of the section, we will discuss the construction of a situation calculus theory ($\mathcal{D}_W$) for each workflow W. Let $\mathcal{D}$ be the set of actions representing the

[4] Indeed, this is the internal representation of a workflow used in our system.

concrete Web services used by the workflow W. For each data node d in W, let v_d be a unique variable representing d. Also, by n_v we denote the integer associated to v.

1. For each concrete service node v in W, let act be the Web service in v and let its representation be $act\,(\mathbf{i}, \mathbf{o})$, then
   - act_{n_v} is an action in $\mathcal{D}_W$.
   - the action precondition axiom of act_{n_v} is

$$Poss\,(act_{n_v}, s) \equiv \bigwedge_{x \in \mathbf{i}} available(x_{n_v}, s)$$

   - for each variable x in $\mathbf{o}$, the conjunct $(a = act_{n_v})$ belongs to the positive precondition effect $\gamma_a^+(x_{n_v}, s)$ of x.
2. For each data node d and service node v in W such that $(d, v) \in E$:
   - Let us assume that there is no link of the form $(., d) \in E$. Let c_d be the constant associated to d and x be an input variable of act, whose value should be bound to c_d as described in the previous section. Then we will add the axiom

$$available\,(x_{n_v}, S_0)$$

     to $\mathcal{D}_W$.
   - Let us consider the case where there are links of the form $(., d) \in E$. Let c_d be the constant associated to d and x be an input variable of act, whose value should be bound to c_{d} as described in the previous section. Then we will add the axiom

$$available\,(x_{n_v}, s) \equiv available(c_d, s)$$

     to $\mathcal{D}_W$.
3. For each data node d and service node v in W such that $(v, d) \in E$, let c_d be the constant associated to d and x be an output variable of act, whose value should be bound to c_d as described in the previous section. Then we will add the conjunct

$$a = act_{n_v}$$

   to the positive precondition effect of the fluent $available\,(c_d, s)$.
4. For each pair of control nodes v and v' such that $(v, v') \in E$, let $x^1, \ldots, x^k$ be the set of outputs of v that are inputs of v'. Then we will add the conjunct

$$a = act_{n_v}$$

   to the positive precondition effect of each fluent $available\,(x^i_{n_{v'}}, s)$.

To complete the construction of the C-Golog program from W, we will need the following generic C-Golog program, denoted by *Transformation_Type*,

$$t_1 \mid t_2 \mid \ldots \mid t_n$$

where t_i's are the available type conversion services. The translation is completed by the following changes to P_W:

- For each abstract service node v in W, let *cond* be the constraint on the service in v and let $S = \{act^1, \ldots, act^k\}$ be the set of services of the type mentioned in v. Then we will replace in P_W l_{n_v} by the C-Golog program

$$Transformation\_Type^* \ (?cond) \ (act^1 \mid \ldots \mid act^k) Transformation\_Type^* \ (?cond)$$

- For each conditional node v in P_W, if there are some inputs $x_{i^1}, \ldots, x_{i^k}$ of v, we will replace l_{n_v} with $l_{n_v}(x_{i^1}, \ldots, x_{i^k})$.

Let P_C be the program obtained from P_W after the changes and $\mathcal{D}_W$ be the situation calculus theory generated from W. We have that P_C is a set of C-Golog programs representing W, whose traces can be computed using an interpreter for C-Golog. These can be computed using any external planner that supports C-Golog [42].

6.7 Executor

6.7.1 Execution Framework

The Web service Description Language (WSDL) is commonly used to describe the functionality of a Web service, i.e., a service that is deployed over the Web and accessed using a communication protocol, such as the Simple Object Access Protocol (SOAP). However, since WSDL is just an interface description language, nothing prevents its use to describe other type of services.

This is exactly the idea behind the *Web Services Invocation Framework (WSIF)* [13], which defines extra binding extensions for software components, like Enterprise Java Beans (EJBs), local Java classes, software accessible over message queues using the Java Messaging Service (JMS) API, and software that can be invoked using the Java Connector architecture. Following this idea, we define additional binding extensions for other relevant classes of services (e.g., Perl scripts, BioPerl modules) and provide a service executor factory with interfaces for registering, removing, and updating service providers with their corresponding WSDL extensions. This is illustrated in Fig. 6.24.

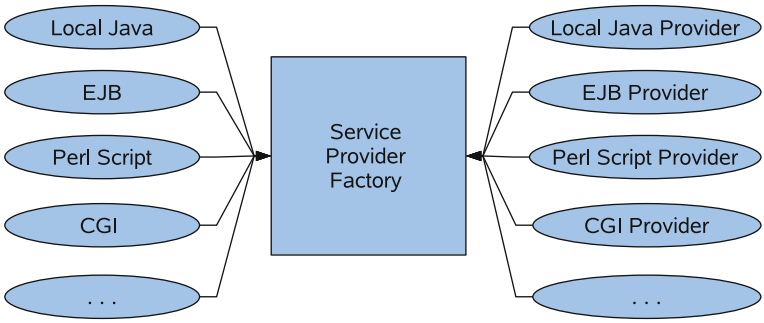


Fig. 6.24 Service provider factory

The execution of the complete workflow is highly parallel. The execution manager will operate on a queue of nodes (see Algorithm 3); each node's execution will determine the successor steps, which will be instantiated as new threads. The ExecutorFactory will create a different executor based on the type of service to be processed (see Algorithm 4). The execution process is monitored with status information for each service/operator node in the workflow. The input/output data of each service/operator node can also be checked at any given point of time during execution (e.g., Fig. 6.25 shows a monitor window during a workflow execution).

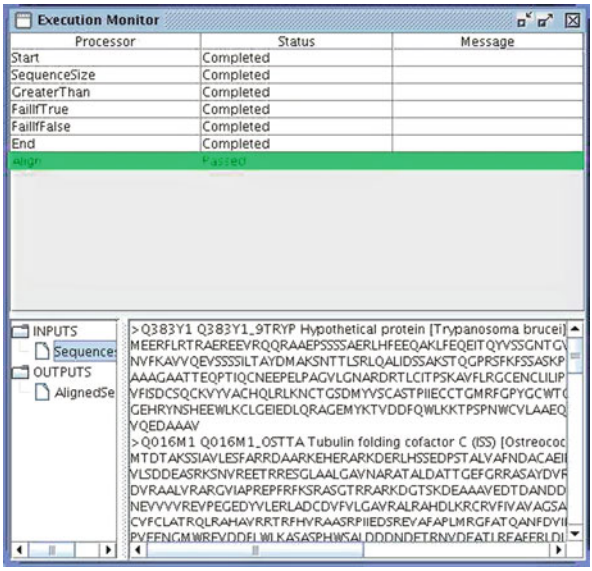


Fig. 6.25 Execution manager

Algorithm 3 Executor

```
Require: Workflow
  queue = new Queue();
  queue.Push(StartNode);
  while (not queue.Empty()) do
    processor = queue.Pop();
    executor = Create New Process Executor from ExecutorFactory for processor
    new Thread(executor).start()
  end while
```

Algorithm 4 Create New Process Executer from ExecutorFactory

```
Require: Processor
  if (Processor is a Web Service) then
    return new SOAPServiceExecutor()
  end if
  if (Processor is an Operator) then
    return new OperatorExecutor()
  end if
  if (Processor is a local Java class) then
    return new LocalJavaExecutor()
  end if
  if (Processor is a control structure) then
    return new ControlStructureExecutor()
  end if
```

An error during the execution of one service node will be propagated through the workflow, so that the whole execution process can be stopped. Furthermore, external programs can be associated with certain data formats so that they can be automatically started for viewing/editing purposes.

6.7.2 Extension Mechanism

An important aspect of BSIS is its open and extensible architecture. This derives from its design, which is based on standard Web services technologies like WSDL and UDDI. In this section, we will overview the mechanisms used to extend the functionalities of the system by incorporating third-party Web services or by developing custom operators. We will also discuss a way of interacting with BioPerl, which is one of the most widely used tools in several areas of bioinformatics (e.g., for phylogenetic analyses).

6.7.2.1 Web Service Providers

In spite of the relatively young age of several Web service technologies, there are already hundreds of bioinformatics Web services deployed by various parties (e.g., Taverna [44] lists over 3,000 services and BioMOBY [48] reports 600 services).

Some relevant providers of services are NCBI [34], EMBL-EBI [23], BioMOBY [48], and the Kyoto Encyclopedia of Genes and Genomes (KEGG) [18].

Ideally, to make use of these service repositories, we would only need to obtain the services' WSDL descriptions, create semantic annotations, and then register them with our registry service. Unfortunately this simple approach will not work, because each service provider has its own XML schema representation for biological data. In order to use these existing Web services, we have to develop wrapper services, taking care of the conversion between a piece of raw data and its XML representation used by the specific service provider.

The full automation can only be achieved if every Web service provider adopts the same XML schema representation (it is hard to have a global schema for every biological data, though) for their biological data. Otherwise, we suggest using the raw data format produced by an application and annotate its meaning in the domain ontology. Web services also need to be registered in some UDDI compatible registry. In this case, BSIS can be configured to query multiple UDDI nodes and to interact with the services through SOAP messages according to the description in their WSDL documents.

6.7.2.2 Custom Operator

Operators are used to customize user operations in the workflow, especially when users require some specialized handling of data which is not available from any existing services. BSIS provides a way for users to define their own operators and load them at run-time, so that they can be used in a workflow. It also provides a way to store these definitions permanently, so that they will remain available for other workflows.

The current implementation only supports custom operators written in Java. In order to make the operator available for use in workflows, two things must be done: first, we need to define the operator in Java – by creating a subclass of the abstract class BsisOperator and then we need to annotate its functionality in XML. An example of the annotation of a new operator (called *SeqListSize*), used to count the number of sequences in a FASTA file, is shown in Fig. 6.26 The annotation describes the input of the *SeqListSize* operator as being of type *ListOfSequence* and its output as being of type *Number*.

```
<operator id="edu.nmsu.cs.bsis.operator.SeqListSize">
    <inputs>
        <input name="Sequences" type="&datatype;#ListOfSequence"/>
    </inputs>
    <outputs>
        <output name="Size" type="&datatype;#Number"/>
    </outputs>
</operator>
```

Fig. 6.26 Annotation for the sample operator

6.7.3 BioPerl Modules

BioPerl [43] is an open source project aimed at providing accessible programming modules to Biological community, which can be used in Perl programs. Currently, BioPerl provides over 800 modules, covering almost every domain of bioinformatics studies. Compared to the use of Web services, BioPerl modules are all installed locally and this enables a more efficient execution. For this reason, we have provided BSIS with the additional capability of transforming a workflow to a BioPerl program that can be executed locally using the available BioPerl modules.

The idea is to annotate each BioPerl module, describing its interfaces and functionalities using the domain ontologies, and creating a template that contains the code for executing a generic task using the APIs from that module. These templates are registered in the BSIS registry with special tags, indicating that they are BioPerl modules.

During the workflow planning phase, the user may indicate his/her preference for BioPerl modules and the planner will search the registry for BioPerl templates satisfying the workflow specification. All BioPerl templates are combined to produce an executable BioPerl program. Fig. 6.27 illustrates this process.

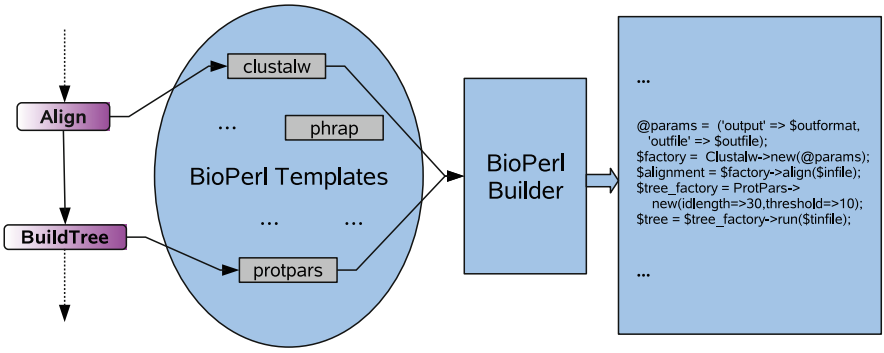


Fig. 6.27 From workflow to BioPerl

6.8 Case Studies and Optimizations

6.8.1 Some Case Studies

A number of use cases have been developed to evaluate the capabilities of the system. All the use cases represent workflows that have been developed by domain scientists assisted by the developers of BSIS. An extensive coverage of these use cases is available in [35]. These use cases cover distinct bioinformatics domains, including phylogenetic studies, gene expression analysis, Gene Ontology (GO) annotations, and protein structure analysis.

Figure 6.28 illustrates a typical workflow in a phylogenetic study. The workflow contains only three abstract service nodes before planning is performed: a *BLAST*

service for the retrieval of relevant sequences, a multiple sequence alignment service, and a phylogenetic tree building service. After planning, these service nodes are instantiated to the NCBI *Blast* service, the *TCoffee* service from EMBL-EBI, and the *ProtPars* service from the PHYLIP package, respectively. Moreover, two additional services have been *automatically* inserted by the planner. One is used to extract from the BLAST report the list of matched sequences before executing the alignment service. The *ReadSeq* service is inserted between *TCoffee* and *ProtPars* to convert the alignment results to the *PHYLIP* format, required by *ProtPars*.

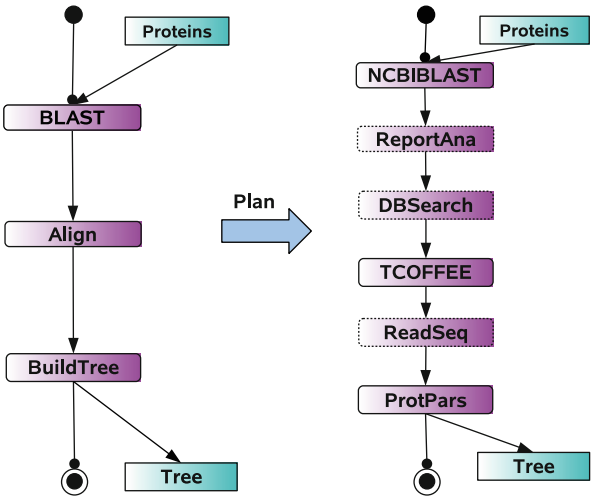


Fig. 6.28 Phylogenetic analysis workflow

Figures 6.29 and 6.30 show other two use cases developed in BSIS. The workflow in Figure 6.29 is in the domain of protein structure analysis: given the primary sequence of a protein, it checks whether the protein contains any trans-membrane regions. If so, two programs are requested to investigate the properties of the trans-membrane region: one draws the hydrophobicity profile and the other draws the propensity profile.

The workflow in Fig. 6.30 describes the process of retrieving the context of a term defined in the Gene Ontology. A term ID is used as input to the workflow. The workflow splits into four branches. The first branch retrieves the ancestors of the term, adding them to the session created by the last branch, and then marking these terms using user-specified colors. The second branch performs a similar task for the immediate children of the input term. The third branch retrieves the parents of the term, then gets the children of the parents (i.e., the sibling of the input term), adding them to the session and coloring them. Finally, these branches merge into a single process, calling *getResults*, to retrieve a text representation of a special graph that can be viewed using *GraphViz*.

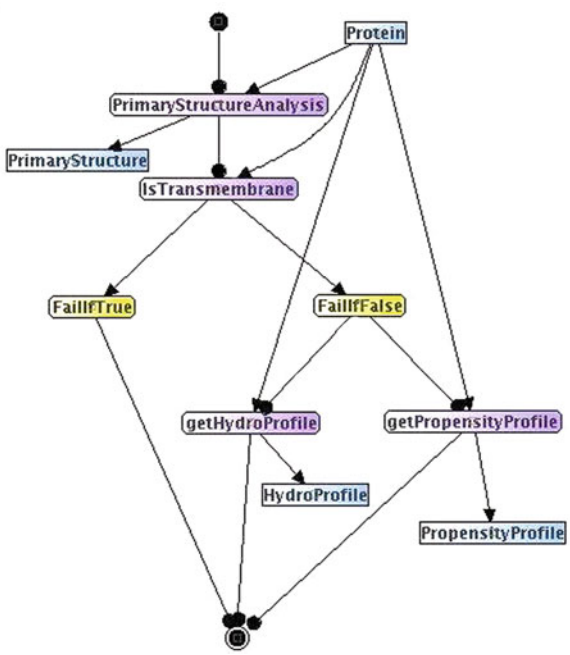


Fig. 6.29 Protein primary structure study

6.8.2 Optimization

The effectiveness of the system depends in a large degree on the efficiency of the planning algorithm, whose running time can be roughly described by the formula

$$A \cdot \underbrace{aC \cdot Q}_{T_G} + R^A \cdot \underbrace{(p(I+O) \cdot P_{s,l} + (1-p)(bI+bO))}_{T_B}$$

where

- A is the number of abstract services
- C is the average number of constraints of a service
- Q is the time required to execute a query to the registry
- R is the average number of instances from the query results of each service
- p is the percentage of data conversion required when performing data binding for a service
- I is the average number of input bindings for a service
- O is the average number of output bindings for a service
- $P_{s,l}$ is the running time for generating a data conversion plan, which is a function of the search space (s) and the average length of the plan (l)

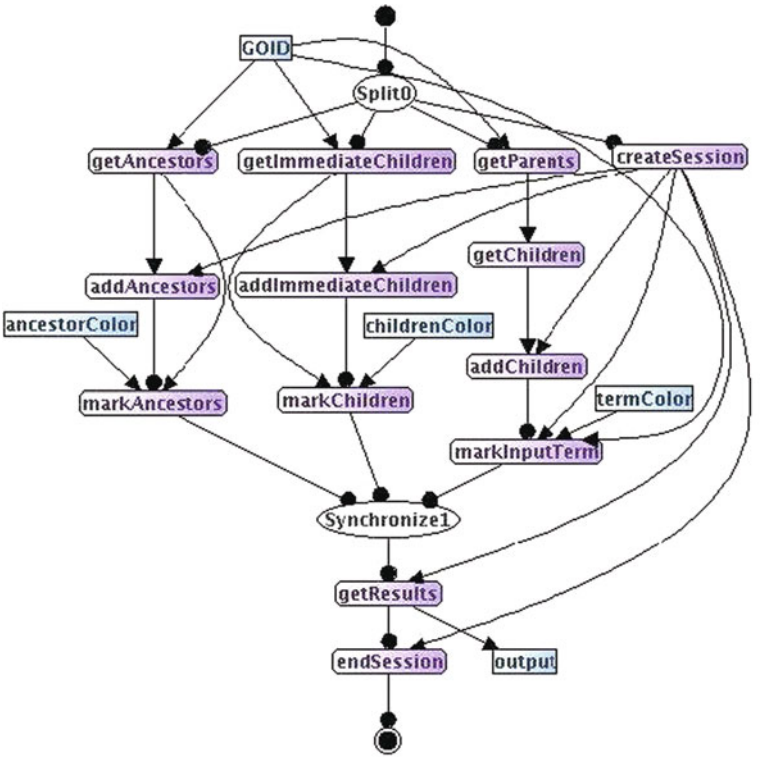


Fig. 6.30 GO context search

- a and b are constant factors related to query processing and data binding, respectively
- T_G represents the time for service mapping
- T_B represents the time for data binding

Figure 6.31 shows a plot which compares the total planning time with the time spend on registry queries during the planning process (these are averages over a collection of use cases).

The line at the top is the total planning time in milliseconds, measured from 12 test cases. The line at the bottom is the total query time during their planning processes. As we can see from Fig. 6.31, a large portion of the planning time is spent on querying (an average of 78.2% for the 12 test cases). Thus, in order to improve plan performance, we should concentrate on decreasing the number of queries.

We have already discussed several methods for plan optimization, including using the Most-Constraint-Service-First heuristics, setting the maximum search depth and defining guided search strategies. The other optimization methods used by BSIS include

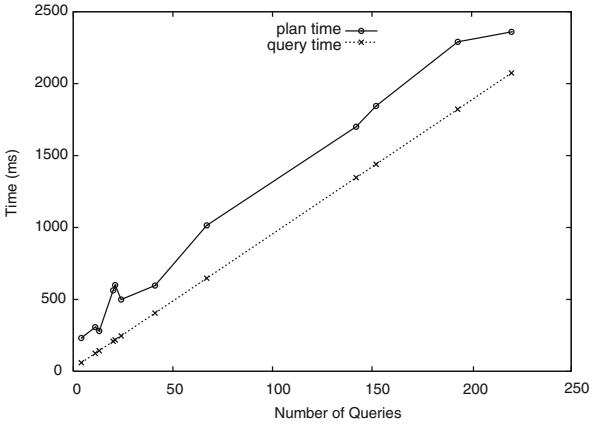


Fig. 6.31 Planning time vs. query time

- *Late interface binding.* This technique is to delay the query for WSDL documents until the data-binding time. The information of a WSDL document will not be retrieved until that service instance is bound to a service node in the workflow. This often helps reducing the number of registry queries to about half of their original number.
- *Data caching.* A local data cache containing mappings between metadata and WSDL components can greatly improve the planning performance. One problem with local data cache is synchronization. Modules can be built to update the data cache entries periodically or a time stamp can be maintained to enable discarding information that is older than a certain age.
- *Improve registry query interface.* The most important improvement regarding the number of queries is obtained by providing a more flexible query interface from a Grimoires registry. For example, if we are going to search a phylogenetic tree building service that uses either a parsimony or a maximum likelihood method, in the current registry model this requires two separate queries, whose results should be merged. By providing support for the *OR* logic within the registry model, a single query could be used to accomplish the same task.

6.9 Conclusion and Future Work

In this chapter, we presented the design and development of the BioService Integration System (BSIS), a computational workbench for bioinformatics. The workbench makes use of Semantic Web and workflow technology to achieve intelligent workflow generation and execution in the domain of bioinformatics investigations. BSIS arguably resolves some of the weaknesses demonstrated by existing state-of-the-art systems for workflow development in bioinformatics domains.

The major contributions of this work can be summarized as follows:

- The design of a workflow language that is capable of representing bioinformatics *abstract plans* of services through the use of metadata.
- The extension of the WSDL model for encoding metadata information from domain-specific ontologies and associating them to Web services in that domain.
- The development of planning algorithms that can be used to convert an abstract workflow into an executable workflow, through the use of dynamic instance binding using the extended WSDL model and the metadata-aware Grimoires registry model.
- The application of the WSIF model to the design of the execution module, using the factory pattern and the extension of the WSIF model for executing local BioPerl modules. A workflow consisting of BioPerl modules can be exported as a BioPerl program for local execution.
- The implementation of a prototype system for the demonstration of feasibility, effectiveness, and efficiency. Also, extensibility of such system has been demonstrated through integrating third-party Web services and custom operators.

The key to the success of such system is the adoption of well-known domain-specific ontologies and the use of such ontologies to annotate services/applications. Additionally, a user-friendly graphical interface, an efficient planning algorithm, and a smooth execution process will speed the acceptance of the system.

The goal of a broader adoption of a workflow infrastructure in a computational science domain requires additional steps to the technical developments illustrated in this chapter. We have identified the following important activities to be performed:

Ontologies: The design we proposed has been created to build on and integrate with existing ontologies for the description of both biologically relevant data and bioinformatics services. Several efforts are in place, at the level of communities of bioinformatics researchers, to create extensive ontologies with broader community acceptance. For example, there is an active community of researchers in evolutionary biology that are working on the creation of a comprehensive data ontology for the field of phylogenetic analysis (the Comparative Data Analysis Ontology – CDAO). It is important to maintain the integration of such emerging ontologies within BSIS, to immediately take advantage of classes of metadata and not exclude any community of potential users. For example, CDAO has already been integrated into BSIS.

Further technical requirements: The developments we described in this chapter provide an effective framework that is usable for several real-life applications. Nevertheless, there are further technical developments that could enhance usability and applicability. The Grimoires registry query interface can be expanded, by building dedicated registry servers that can provide interfaces for service registration and querying. It is also important to expand the creation of wrappers to enable the use of well-known bioinformatics applications as Web services.

Non-technical requirements: It is vital to expand the cooperation with field practitioners (e.g., biomedical researchers, bioinformaticians) to create workflows for typical bioinformatics studies in different domains, like phylogenetic analyses, protein structure prediction, and microarray analyses.

Acknowledgments The authors would like to thank the anonymous reviewers for their insightful comments. The authors wish to thank the following researchers for helping in different stages of this project: Gopal Gupta, Arlin Stoltzfus, Julie Thompson, Francisco Prosdocimi, Brook Milligan, Tu Phan, and Samat Jain.

The research has been partially supported by NSF grants IIS-0812267, HRD-0420407, and CNS-0220590.

References

1. Altintas, I., Berkley, C., Jaeger, E., Jones, M., Ludscher, B., Mock, S.: Kepler: An extensible system for design and execution of scientific workflows. In: Proceedings of the16th International Conference on Scientific and Statistical Database Management (SSDBM'04), IEEE Computer Society, Los Alamitos, CA (2004) 21–23
2. Ankolekar, A., Burstein, M., Hobbs, J., Lassila, O., Martin, D., McIlraith, S., Narayanan, S., Paolucci, M., Payne, T., Sycara, K., Zeng, H.: DAML-S: Semantic markup for web services. In: Proceedings of the International Semantic Web Working Symposium (SWWS) (2001)
3. Balzer, S., Liebig, T., Wagner, M.: Pitfalls of OWL-S: A practical semantic web use case. In: International Conference on Service Oriented Computing, ACM Press, New York, NY (2004)
4. Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D., Patel-Schneider, P., Stein, L.A.: OWL Web Ontology Language Reference. Technical Report, W3C (2004)
5. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Scientific American, Harper, San Francisco, CA (2001)
6. Chapman, B., Chang, J.: BioPython: Python tools for computational biology. ACM SIGBIO Newsletter 20(2) 15–19 (2000)
7. Christensen, E., Curbera, F., Meredith, G., Weeravarana, S.: Web Services Description Language (WSDL) 1.1. Technical Report, W3C (2001)
8. Clement, L., Hately, A., von Riegen, C., Rogers, T.: UDDI Version 3.0.2. Technical Report 20041019, OASIS (2004)
9. Consel, C.: Domain specific languages: What, why, how. Electronic Notes in Theoretical Computer Science 65(3) (2002)
10. De Giacomo, G., Lespérance, Y., Levesque, H.: *ConGolog*, a concurrent programming language based on the situation calculus. Artificial Intelligence 121(1–2) (2000) 109–169
11. Erl, T.: Service-Oriented Architecture: Concepts, Technology, and Design. Prentice Hall, Upper Saddle River, New Jersey (2005)
12. Fernández-Olivares, J., Garzón, T., Castillo, L., Garcia-Pérez, O., Palao, F.: A middle-ware for the automated composition and invocation of semantic web services based on temporal HTN planning techniques. In: Conference of the Spanish Association for Artificial Intelligence (CAEPIA). Springer, New York (2007) 70–79
13. Fremantle, P.: Applying the Web services invocation framework. IBM DeveloperWorks, Technical Report, IBM, Armonk, NY (2002)
14. Gonnet, G.H., Hallett, M.T., Korostensky, C., Bernardin, L.: Darwin v. 2.0: An interpreted computer language for the biosciences. Bioinformatics 16 101–103 (2000)
15. Holmes, I.: Use cases for GNU Make in Bioinformatics Analyses. Technical Report, Bio Wiki (2007). biowiki.org/MakefileLabMeeting
16. Hull, D., Wolstencroft, K., Stevens, R., Goble, C., Pocock, M., Li, P., Oinn, T.: Taverna: A tool for building and running workflows of services. Nucleic Acids Research Web Services Issue (2006)
17. Janies, D.A., Wheeler, W.: POY version 3.0, Documentation and Command Summary, Phylogeny Reconstruction Via Direct Optimization of DNA and Other Data. Technical Report, BMI Technical Report:OSUBMI-TR-2002-n03 (2002)

18. Kanehisa, M., Goto, S., Hattori, M., Aoki-Kinoshita, K., Itoh, M., Kawashima, S., Katayama, T., Araki, M., Hirakawa, M.: From genomics to chemical genomics: New developments in KEGG. Nucleic Acids Research 34 354–357 (2006)
19. Katayama, T., Nakao, M.C., Goto, N., Tanaka, N.: Bioruby+chemruby: An exploratory software project. GIW 2004 Poster Abstracts, S06 (2005)
20. Kim, H.S., Kim, I.C.: Mapping semantic web service descriptions to planning domain knowledge. In: World Congress on Medical Physics and Biomedical Engineering (2006) 388–391
21. Kosar, T., López, P.M., Barrientos, P., Mernik, M.: A preliminary study on various implementation approaches of domain-specific languages. Information and Software Technology 50(5) (2008)
22. Kumar, S., Dudley, J., Nei, M., Tamura, K.: MEGA: A biologist-centric software for evolutionary analysis of DNA and protein sequences. Briefings in Bioinformatics 9 (2008)
23. Labarga, A., Valentin, F., Andersson, M., Lopez, R.: Web Services at the European Bioinformatics Institute. Nucleic Acids Research Web Services Issue (2007)
24. Levesque, H., Reiter, R., Lesperance, Y., Lin, F., Scherl, R.: GOLOG: A logic programming language for dynamic domains. Journal of Logic Programming 31(1–3) (1997) 59–84
25. Lord, P., Wroe, C., Stevens, R., Goble, C., Miles, S., Moreau, L., Decker, K., Payne, T., Papay, J.: Personalised grid service discovery. In: Proceedings of the UK OST e-Science Second All Hands Meeting 2003 (AHM'03), WarwickPrint, University of Warwick, UK (2003)
26. Luo, J., Montrose, B., Kim, A., Khashnobish, A., Kang, M.: Adding OWL-S Support to the Existing UDDI Infrastructure. IEEE International Conference on Web Services (ICWS'06), IEEE Computer Society, Los Alamitos, CA (2006) 153–162
27. Maddison, W.P., Maddison, D.: Mesquite: A modular system for evolutionary analysis. Version 1.05. URL http://www.umanitoba.ca/afs/plant_science/psgendb/doc/mesquite/doc/manual.html. Accessed 21 Jul 2010
28. Maximilien, E., Singh, M.: Reputation and endorsement for web services. ACM SIGecom Exchanges 3(1) (2002) 24–3
29. McIlraith, S., Son, T.: Adapting golog for composition of semantic web services. In: Proceedings of the Eighth International Conference on Principles of Knowledge Representation and Reasoning (KR'2002), Morgan Kaufmann, San Francisco, CA (2002) 482–493
30. Miller, J., Verma, K., Rajasekaran, P., Sheth, A., Aggarwal, R., Sivashanmugan, K.: WSDL-S: Adding Semantic to WSDL. Technical Report, LSDIS Lab, University of Georgia (2004)
31. Moene, I., Subramaniam, S., Darin, D., Leergaard, T., Bjaalie, J.: Toward a workbench for rodent brain image data: Systems architecture and design. Neuroinformatics 5(1) (2007) 35–58
32. Moreau, L., Miles, S., Papay, J., Decker, K., Payne, T.: Publishing Semantic Descriptions of Services. Technical Report, Global Grid Forum (2003)
33. Mungall, C.: BioMake: Specifying Biocompute Pipelines with Makefiles and Beyond. Technical Report, Berkeley Drosophila Genome Project (2006)
34. NCBI: Entrez Help (2006). URL http://www.ncbi.nlm.nih.gov/books/bv.fcgi?rid=helpentrez.chapter.EntrezHelp. Accessed 21 Jul 2010
35. Pan, Y.: Automating Bioinformatics Tasks Through Intelligent Workflow Construction. Ph.D. Thesis, New Mexico State University, Las Cruces, NM (2007)
36. Papazoglou, M., Dubray, J.J.: A Survey of Web Service Technologies. Technical Report DIT-04-058, University of Trento, Italy (2004)
37. Parker, D., Gorlick, M., Lee, C.: Evolving from bioinformatics in-the-small to Bioinformatics in-the-large. OMICS 7(1) (2003) 37–48
38. Parkinson, J., Anthony, A., Wasmuth, J., Schmid, R., Hedley, A., Blaxter, M.: PartiGene - constructing partial genomes. Bioinformatics 20(9) (2004) 1398–1404
39. Pocock, M., Down, T., Hubbard, T.: BioJava: Open source components for bioinformatics. SIGBIO Newsletter 20(2) (2000) 10–12. DOI http://doi.acm.org/10.1145/360262.360266

40. Qiu, W., Schisler, N., Stoltzfus, A.: The evolutionary gain of spliceosomal introns: Sequence and phase preferences. Molecular Biology Evolution 21(7) (2004) 1252–1263
41. Ran, S.: A model for web services discovery with QoS. ACM SIGecom Exchanges 4(1) (2003) 1–10
42. Reiter, R.: Knowledge in action: Logical foundations for describing and implementing dynamical systems. MIT Press, Cambridge, MA (2001)
43. Stajich, J., Block, D., Boulez, K., Brenner, S., Chervitz, S., Dagdigian, C., Fuellen, G., Gilbert, J., Korf, I., Lapp, H., Lehvslaiho, H., Matsalla, C., Mungall, C., Osborne, B., Pocock, M., Schattner, P., Senger, M., Stein, L., Stupka, E., Wilkinson, M., Birney, E.: The BioPerl toolkit: Perl modules for the life sciences. Genome Research 12(10) (2002) 1611–1618
44. Stevens, R., Robinson, A., Goble, C.: *my*Grid: Personalised bioinformatics on the information grid. Bioinformatics 19(1) (2003) 302–304
45. Stollberg, M., Roman, D., Gomez, J.: A mediated approach towards web service choreography. In: Proceedings of the Workshop "Semantic Web Services: Preparing to Meet the World of Business Applications" Held at the 3rd International Semantic Web Conference, IEEE Computer Society, Los Alamitos, CA (2004)
46. Tamura, K., Dudley, J., Nei, M., Kumar, S.: MEGA4: Molecular evolutionary genetics analysis (MEGA) software version 4.0. Molecular Biology and Evolution 24 (2007) 1596–1599
47. Thompson, J., Higgins, D., Gibson, T.: CLUSTALW: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting, positions-specific gap penalties and weight matrix choice. Nucleic Acids Research 22 (1994) 4673–4680
48. Wilkinson, M., Links, M.: BioMOBY: An open-source biological web services proposal. Briefings in Bioinformatics 3(4) (2002) 331–341
49. Yu, B., Singh, M.: An evidential model for distributed reputation management. In: Proceedings of AAMAS'02, ACM Press, New York, NY (2002)

Chapter 7
Near-Miss Detection in Nursing: Rules and Semantics

Mikhail Simonov and Flavia Mazzitelli

Abstract Nursing science deals with human-to-human interaction delivering care service, showing several peculiarities compared with other life science domains. Semantic technologies can be applied to clinical nursing, where the risk management forms a particular application class. Bone marrow transplantation is a specific sub-domain in which nursing process lacks the quality of service's predictability at run time, requiring error detection before their happening and semantic technologies complementing best nursing strategies. The intelligent mix of technologies delivering proactive feedback to human actors in a natural way is a challenge. We investigate on possible risk control measures in the above-said nursing, suggesting a combination of ICT and knowledge technologies.

7.1 Introduction

The Semantic Web researchers have largely focused on formal aspects of logic languages or general-purpose semantic application development. Several attempts to apply the knowledge technologies to life sciences are known, especially in the clinical care domain, supporting the doctor's activities in various forms, ranging from the simple automation of the clinical guidelines to the sophisticated real-time decision support systems offering advanced mining and data-warehousing features. Several projects, including IKF (E!2235) and DOC@HAND (FP7), have delivered their outcomes in this area, and many publications treat different topics characterising the life science with its sub-disciplines. Almost all authors note the complexity and highly structured nature of the topic. However, one of the largest sub-domains is not sufficiently explored by knowledge technologies.

M. Simonov (✉)
ISMB, Via P.C. Boggio 61, 10138 Turin, Italy
e-mail: simonov@ismb.it

H. Chen et al. (eds.), *Semantic e-Science*, Annals of Information Systems 11,
DOI 10.1007/978-1-4419-5908-9_7,

Modern clinical nursing is a highly qualified professional activity aimed to deliver care service to patients, requiring an amount of the knowledge comparable with that of the doctors. Nursing informatics is a new field of specialist nursing practice, integrating information and computer science with nursing to enable nurses to collect, to process, and to manage data. Nowadays the university degree is a requirement to start the professional nursing; consequently the knowledge base to work with is ample and well differentiated, depending on the selected specialisation. The further level of complexity is the human-to-human interaction, difficult to formalise and to describe, even if the quality systems try to release ruling clinical guidelines. Among several nursing specialisations there is one particular sub-domain – the bone marrow transplant nursing (*BMT* hereafter) – presenting specificities, offering a good challenge to build an exemplification of how the knowledge technologies might be applied for.

Bone marrow stem cell transplantation is a medical procedure in the fields of haematology and oncology manipulating the blood stem cells derived from the bone marrow. It is most often performed for people with diseases of the blood, bone marrow, or certain types of cancer. Bone marrow transplantation usually requires that the recipient's own bone marrow is destroyed, so patients may stay for several weeks with low counts of white blood cells, putting them at high risk of infections and sepsis, with the consequent treatment-related mortality (please refer Section 7.2.2. for more details). BMT nursing also protects against the bacteria being introduced.

To manage the possible infections, the nursing activities should be formalised and described in the risk-related terms, making possible the observation of the real life events associated with the risks and their automated triggering by intelligent computers. The challenge is the capability to process data, to elicit knowledge, and to make knowledge-based decisions about patient care. The semantic technology applied to clinical nursing generates a significant added value because of enabling of the knowledge-supported reasoning with efficient risk management features.

The nursing activities are complex, but composed of several atomic elements, showing some recurrent patterns. The real-time reasoning about the manifestation of certain events might deliver the important indications about the maturation of the risks or about the normality.

From a service point of view, nursing might be defined as "service", and it is reasonable to speak about the quality of service and the service quality predictability; however, the toolkit for the automation of nursing activities is significantly different from the instruments traditionally used in Semantic Web applications targeting general practitioners, doctors, or researchers in life sciences.

This chapter presents how the semantic (knowledge) e-Science might be applied to BMT nursing, a sub-domain of life sciences, in which it is essential to detect the risky condition before entering the patient room because of the associated mortality: immune system is depressed totally to allow the transplantation. The risk management in BMT nursing aims to detect a near miss instead of an error; it

reasons and takes the decisions very fast in order to advert human actors proactively in real time. The combination of technologies includes RFID, wearable sensors, process modelling, domain formalisation (ontology), Internet of Things, context awareness, plus the knowledge technology aspects such as Fuzzy rules for errors and plan recognition systems. We accompany the illustration by a possible prototype embodiment.

The approach adopted by the authors is the phenomenological study of the BMT nursing activities, permitting to describe them as processes and the composition (superposition) of the elementary processes. The above-mentioned study reveals the possible nodes where the risks might happen in real life and correlates the findings with the taxonomy of the erroneous actions proposed by Hollnagel [37]. Authors observe the impact of the different possible compositions, comprising the wrong ones, the correct and incorrect timing sequences, and formalise the specific auxiliary processes routinely present in any BMT nursing operation. The investigation about the timing properties, the risk maturation, and the decision trees indicates the possible tools capable to manage the real life processes in BMT nursing. Authors discuss the combination of the different technologies, making possible the semantic mining and the erroneous action's discovery from within the BMT care processes.

The work starts from the overview of the nursing domain and its specificities; introduces the near-miss concept and discusses the human factor; and speaks about the services, events, their ordering, and the possible formalisation of the behavioural patterns. The technology-related part gives an overview of the identification and sensing techniques monitoring the nurse–patient interactions and introduces the knowledge components, Fuzzy systems, and rule-based reasoning. The nursing process modelling is the most significant part because it shows how to exploit the nomothetic components and apply the description–situation design patterns.

The main advantage of the adopted method is the simplicity, extremely beneficial in the knowledge transfer terms: the focus group – including the nurses – discloses the essential domain knowledge because of the complete understanding and participation in the research activities. The meaningful Fuzzy rules add value because of explaining clearly what is happening or what might happen in the near future, while the reasoning made over the well-formalised auxiliary processes instead of the complex and variable main activities ensures the calculability of the result.

7.2 Nursing Domain and Near Miss

Nursing is a profession concerned with the provision of services essential to the maintenance and restoration of health by attending to the needs of sick persons. Another definition of nursing is a human practice discipline that facilitates well-being using a scientific knowledge base and values in a caring relationship.

A near miss is an unplanned event with harmful potential that does not result in injury, illness, or damage, because the break in the event chain prevented the fatality, in which the human error is an initiating event. The domain knowledge in machine-processable representation becomes essential to set up the computerised support and manage near miss.

7.2.1 Nursing

Patient care implying the repetitive *human-to-human interactions* is a part of nursing. Nurses use the nursing process to assess, plan, implement, and evaluate patient care, protecting also patients from the external additional risk factors. The critical thinking, the best practices, and the application of evidence-based concepts to patient problems in a particular setting are tools used by nurses, offering to ICT a challenge to add some value to the above-mentioned human-to-human processes. Nurses deal with patients, their health records, including electronic ones, clinicians; interact with artefacts enabling to take vital signs, vital parameters; account the patient's perception about the events; and sum up both objective and subjective findings. The sources of the nursing knowledge are multiple: books, documents, clinical guidelines, description of the best practices, clinical cases, etc.

The responsibilities of the nurse include appropriate use and understanding of advanced technology and sufficiently complex interventions, as well as the ability to assess and monitor conditions against the deterioration, capitalising on the skills to act appropriately. Several real life situations requiring immediate and appropriate responses can be emotionally and mentally draining.

Nurses dealing with patients often experience episodes of *uncertainty* characterised by the *unpredictability* of events, recurrences, and exacerbation. The unpredictability emerges frequently on the patient deterioration – due to the nature of the disease, *busyness* – due to the workload and intensity of the shift, and weariness – due to the circadian rhythm or the long-lasting shifts, illustrating also how nursing can affect the delivery and quality of care. We see that the unpredictability is a common thread. The busyness impacts negatively on the delivery and quality of patient care, causing stress, but because of the uncertainty and intensity of the busyness, this contributes to unpredictability.

The role of the ICT and semantic technologies cannot be underestimated in this domain of life science, because of the computer support offering predictability in certain well-formalised situations; however, the tacit knowledge being a most important component of the decision-making in nursing, the task is complex. The tacit knowledge supported by long clinical experience, the most difficult to find, capture, and elicit, and the human factor in decision process are main elements helping to solve the uncertainty and the unpredictability of the nursing service.

The ICT problem of unpredictability of software services appears explored, while the theme of unpredictability in nursing is new. We find in [1] the concept of uncertainty in illness, a proposition of an alternate model of uncertainty in the illness experience, and characteristics of the illness linked with the situation – ambiguity,

vagueness, unpredictability, unfamiliarity, inconsistency, and lack of information. Three attributes of the concept of uncertainty, identified as probability, temporality, and perception, become elements to embed in conceptual knowledge modelling.

Hilton [2] describes uncertainty as a cognitive state created when an event cannot be adequately defined or categorised owing to lack of information. In order to organise information, a person must be able to recognise and classify it. This requires that the stimuli be specific, familiar, consistent, complete, limited in number, and clear in boundaries. ICT system equipped by appropriate sensors, sufficient to account all such stimuli happening in a real world, seldom has required characteristics to react in a meaningful way, and therefore the unpredictability acquires the ICT meaning. The proposition cannot be universal since the speed of the change in patient status in different clinical settings is unpredictable as well, depending on the care environment.

ICT offers nursing an opportunity to automate the care activities, helping nurses and patients to access the information they need. Moreover clinical judgement involves decision-making, and nurses, like doctors and other health professionals, use a decision-making process assessing the patient, identifying his/her problems, planning and implementing appropriate care interventions, and evaluating the results. Semantic technologies can provide much more: sophisticated situation-aware and decision support systems. In the patient management, the automated monitoring of physiological signs enables nurses to care for more patients and increase the efficiency. The step beyond is the adoption of the knowledge technologies, enabling to assess the situation, to consult the formalised knowledge, and to elaborate the intended action, which might be used directly. The intended action might be compared with the undertaken one, enabling the risk management automation, including the near-miss detection. The knowledge technologies combat the uncertainty and support the decision-making.

7.2.2 Bone Marrow Transplantation

A haematopoietic stem cell, or bone marrow transplantation [3], may be used in patients with leukaemia, lymphoma, and some other tumoral forms. Bone marrow contains stem cells producing normal blood ones, but sometimes a number of defective, cancer cells. High-dose chemo- or radiotherapies destroy such cancers. However, such treatments damage normal cells too and compromise the capacity to combat external agents, making BMT recipients extremely vulnerable in terms of possible infections, because of lowest levels of blood counts. After the anti-tumour therapy, healthy bone marrow is administered by transplantation to restore normal stem cells life cycle. The chemotherapies help also to avoid the rejection risk, e.g. BMT being destroyed by healthy immune system before functional recovery. In a successful BMT, the new stem cells migrate and engraft, starting new normal blood cells' production cycles.

Prior to BMT, a number of tests ensure that the patient is physically able to undertake the intervention, identifying and treating proactively any potential problem. The

tissue-matching human leucocyte antigen test in both donor and recipient identifies and compares antigens, antibody production markers. Central venous catheter is applied prior to BMT to allow the direct and needleless delivery of nutrition, therapies, or other elements into the bloodstream, because frequent punctures increase the infection risk. Colony-stimulating factors might be subcutaneous or intravenously injected helping white blood cells recovery after chemotherapy, reducing risk of infection, and growing stem cells. Bone marrow harvesting in the operating room is ensured by general anaesthesia.

The anti-tumoral therapy is the second stage of the care process adopting very high-dose chemo combined sometimes with radiation to destroy stem and blood cells, preparing the body to become BMT recipient. Intravenous fluids accompanying the chemotherapy contribute to minimise induced damages, while special drugs control side effects. The patient quickly losing white blood cells becomes very fragile. The *ineffective immune system* cannot protect from infections, requiring the patient being isolated in special hospital room until the new bone marrow begins to grow and the immune system restarts. The nursing and care in this stage, undertaken in a strictly *isolated controlled environment* [4], are strictly ruled by precise guidelines.

The bone marrow infusion followed by normal blood cells production starting after several weeks complete the therapy. The immune system is going to recovery, while risks are going to disappear. An instrument to ensure the fulfilment of the rules during this period capable to manage the near miss can be enabled by knowledge technology.

7.2.3 Bone Marrow Transplantation Nursing

The in-hospital stay of patients between the chemotherapy [5] and the effective immune system recovery is associated with nosocomial infection risks [6], making important the risk prevention in BMT nursing. Hospital-acquired infection is still a problem, challenging new applications. The patient room and atmosphere are separated from the rest of the department by two doors [7]. Caregivers accessing patient rooms undertake accurate measures ensuring the absence of pathologic agents being the main risk in such an environment [8]. A sink is typically installed between two doors allowing hand washing, prior to applying clean gloves. Typical near-miss situation is a tentative entry in the patient's room under the non-verified sterility condition. Nursing techniques are the tacit knowledge to learn, convertible into machine-processable heuristics. The staff is professionally prepared and motivated, making possible rules collection by engineers using usual tools.

ICT support in BMT nursing might be the introduction of some *sensorial devices* capable of detecting the conditions characterising risks, plus an engine running encoded and learnt heuristics in order to detect and trigger any dangerous situation. This allows raising the proactive action: to warn the personnel falling into near-miss condition, otherwise going to introduce the risk. The engineering solution might combine electric/electronic tools (sensors, actuators), knowledge technology

(intelligent heuristics, context and situation awareness, and machine reasoning), and ICT (process monitoring and event triggering) with advanced human–machine interfaces. Managing risks in nursing remains a complex task requiring several technologies being mixed.

7.2.4 Human Factor and Near Miss

Human error has been extensively studied by psychologists such as Reason [9] and Norman [10], distinguishing two ways in which action can logically fail to achieve its purpose: a mistake defined as an error of judgement or inference, where the action goes as *planned* but where the plan is incorrect, and a slip defined as an action that is not in accord with the actor's intention. In other words, a mistake is a failure during plan generation, while a slip is a failure at run time of a good plan poorly *executed.*

Norman [11] points out that the consequences of a mistake are usually more severe than those of a slip. He argues that humans invented cognitive artefacts to overcome limitations in their memory and processing capabilities: pen and paper, chalk/white-boards, mathematics and logic, and slide rules. We add interactive software tools, and – in particular – artificial intelligence planning systems.

Psychological research has concentrated on studying erroneous action. Hollnagel [12] speaks about two fundamentally different ways to consider erroneous actions: one is with regard to their phenotype – *how slips appear* when expressed in actions, the other is with regard to their genotype, e.g. the functional characteristics or mechanisms of the human cognitive system that are assumed to be a contributing cause of slips. One way of improving interactive software tools is to incorporate functionality that can recognise possible erroneous actions and alert the user to their occurrence. From this point of view it is sufficient to be concerned only with the phenotypes. He avers that, provided patterns of plans and rules for deviations are given, it is a relatively simple matter to construct a plan recognition system. However, erroneous actions are not completely random events, but depend on the context and specifics of the user's current task. Knowledge of the genotype, suitably combined with context information, may aid in improving the speed and correctness of recognition and classification of deviations.

The CNSI taxonomy [13] contains several mechanisms of human malfunction (i.e. genotypes): discrimination, input information processing, recall, inferences, and physical co-ordination. Reason [14] proposed the influential Generic Error-Modelling System (GEMS) model that relates typical error types with error-shaping factors, using Rasmussen three-level theory of cognitive human performance. These mechanisms and factors [15] are of limited interest unless they can be related to a classification of error types, i.e. phenotypes.

Hollnagel has found a coherent description of the phenotypes, derived partly from an analysis of the finite number of ways in which a set of actions can be permuted and partly from the concepts provided by the CNSI and GEMS models. His starting point is that purposeful action is guided by goals and carried out according to plans. To produce an initial characterisation of the simple phenotypes

of erroneous actions, Hollnagel considers a plan as a *linear sequence of actions* leading to a goal. He assumes that the action sequence is necessary and sufficient to reach the goal. Two additional simplifying assumptions are that the actions take place in an ordinal sequence, i.e. neither the time of their occurrence nor their duration is of importance, and the single actions can be executed one at a time with success or failure of each action being determined immediately. Figure 7.1 depicts the simple phenotypes of erroneous action: in each case the correct action sequence is (A, B, C, D, E). From these simple phenotypes, he constructs the taxonomy of phenotypes of erroneous action then.

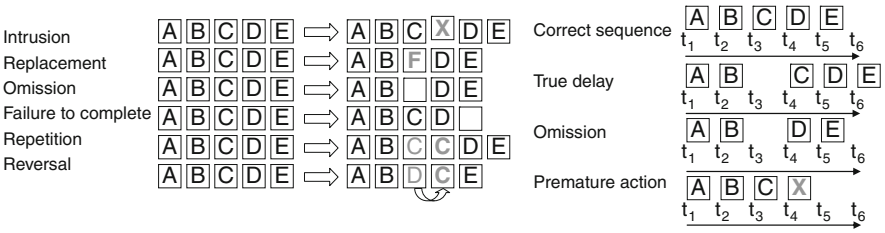


Fig. 7.1 Hollnagel simple phenotypes

In risk management area an efficient solution might detect – but should not document – erroneous actions to permit the improvements of the practice through human interventions. However, the preventive action would be desirable, before an error happens, introducing the near-miss concept. We define near miss as a rule violation detected before happening; however, many harmful situations remain unobservable in real life systems, missing a detection tool or lacking formalised rules. An answered mobile phone while nursing in BMT domain becomes an intrusion element capable to break the sterility chain. The omission of the pre-infusion procedure might compromise the therapy. The forgotten post-infusion venous line cleaning is a failure to complete. The mask and gloves wearing procedures might show the reversal phenotype. Proactive help suggesting the repetition of the hand washing procedure and the pre-infusion and post-infusion completion can be set up.

Authors start from the above-mentioned taxonomy of erroneous actions to build the computerised system capable to detect near miss in nursing and raise an event permitting to break the fatal event chain, but – contrary to Hollnagel – authors consider vital the fact that the actions taking place generate an event sequence, in which the time of their occurrence and their duration are essential to understand semantics and react appropriately. The main difference is the time reasoning permitting to discover the abnormality patterns in actions.

7.2.5 Service, SLA, Plans, Rules, Patterns

Nursing configures special measures to avoid and manage risks. Those are formalised by clinical guidelines and reflected by nursing practice, forming the tacit

knowledge. BMT nursing might be abstracted as a service governed by service level agreements (*SLA* hereafter). BMT nursing becomes a particular class of healthcare services, requiring precise quality of service (*QoS* hereafter) levels to be guaranteed at run time, because in the case of the nosocomial infections the mortality becomes high. It is a complex human-to-human interaction, and being abstracted as "service" it will lack QoS predictability at run time. The late detected SLA violation is an error, which becomes a near miss being detected before happening or passes unobservable missing a mechanism to detect SLA. In the first two cases the quality is managed, consequently an efficient risk management should enable to detect the possible SLA violation before it will happen, ensuring the QoS predictability at run time.

BMT nursing is decomposable in several atomic processes, those described in well-formalised guidelines, simplifying the knowledge modelling. There are several recurrent patterns, which might be observed and modelled using directional processes. An entry in the patient room is an example showing the superposition of several items, such as opening doors, washing hands, using masks, gloves. However, the BMT nursing knowledge is tacit and not directly machine-processable.

7.2.6 Real Life Event Ordering

The real life shows a set of events originated by actors happening in time and space. The spatial and temporal relations among them characterised by specific properties can be fixed using predicates. Let us observe two relations: an event S_1 happening *nearby* (***N***) or *prior to* (***P***) event S_2. For any ($\forall$) event S we cannot say that S happen *nearby* S, and S happen *prior to* S. Both binary relations ***N*** and ***P*** are anti-reflexive (1), while ***P*** is transitive (2).

$$\forall S \leftarrow (S\,\boldsymbol{N}\,S) \;\text{ AND }\; \forall S \leftarrow (S\boldsymbol{P}S) \tag{1}$$

$$\text{IF}(S_1\,\boldsymbol{P}\,S_2)\,\text{AND}\,(S_2\,\boldsymbol{P}\,S_3)\,\text{THEN}\;\;(S_1\,\boldsymbol{P}\,S_3) \tag{2}$$

The temporal relation ***P*** is transitive, but the spatial one is not. The anti-reflexive and transitive binary relations might create a partial ordering, so temporal relation ***P*** orders the events in time. We exploit this aspect to assess an event chain happening in real life and detected by sensors, contrary to other application classes ignoring the difference between the spatial ***P*** and the temporal one. M. Eliade [16] distinguishes between an infinite cyclic time with the event chain repeated an infinite number of times and a limited cyclic time where the number of repetitions is finite. However, the temporal relation ***P*** can be cyclic or ordering, but not both. Observing three events A, B, and C happening cyclically (Fig. 7.2), we can speak about (A ***P*** B), (B ***P*** C), (A ***P*** C), but cannot discriminate the first event.

Knowing the timing only – event A happening at 5:00, B happening at 7:00, and C happening at 11:00 – we can show events using geometric representations (Fig. 7.2), but cannot assess their ordering lacking the date. We model the timing

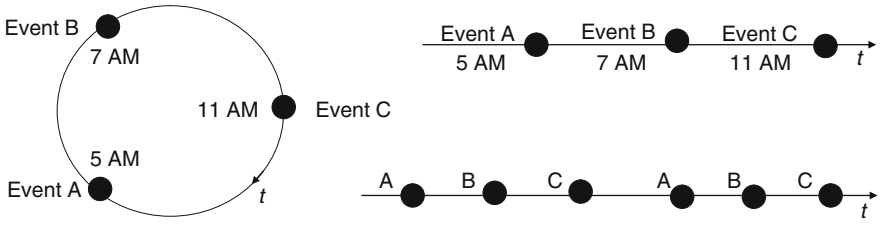


Fig. 7.2 Cyclic events

using a spatial structure, such as a line or a circle, which appears valid until the events are ordered. However, the points A, B, and C, being shown on the circle, violate the ordering axioms. Using non-ordered discrete points A, B, C, . . ., A, B, C along an infinite time arrow we cannot model the time, even if each event chain [A, B, C] is already ordered. The cyclic event chain can be assessed fixing the initiating event permitting collating the sequence correctly: the time needs to be ordered to help the reasoning about the events. Physically (S_1 $\boldsymbol{P}$ S_2) means that S_1 and S_2 are connected by temporal segment. S_1 happening contemporary ($\boldsymbol{C}$) with S_2 means that S_1 and S_2 are connected by spatial segment. In other words we say (S_1 $\boldsymbol{P}$ S_2) if S_1 can undertake a physical action on S_2, and (S_1 $\boldsymbol{C}$ S_2) if S_1 cannot act physically on S_2 and S_2 cannot act on S_1. Following Reichenbach [17], the contemporaneity $\boldsymbol{C}$ is reflexive (3), C is symmetric (4), and C is transitive (5).

$$\text{For} \forall S (S\, C\, S) \tag{3}$$

$$\text{For } \forall S \text{ and} \forall S' (S\, \boldsymbol{C}\, S') \rightarrow (S'\, \boldsymbol{C}\, S) \tag{4}$$

$$\text{For } \forall S, \forall S' \text{ and} \forall S'' (S\, \boldsymbol{C}\, S') \text{AND} (S'\, \boldsymbol{C}\, S'') \rightarrow \left(S'\, \boldsymbol{C}\, S'\right) \tag{5}$$

Any relation showing reflexive, symmetric, and transitive properties is the equivalence. However, in the special relativistic theory the reflexive and symmetric relation $\boldsymbol{C}$ is not transitive. Let us observe an example: someone enters the patient room (event S_1) bringing some bacteria and the BMT patient develops some infection (event S_2). Because of the cause–effect relation we model (S_1 $\boldsymbol{R}$ S_2), reasoning about the nature of $\boldsymbol{R}$ becomes possible to detect and classify an unwanted pattern, handling also near miss if any. In not-empty sets any equivalence defines a partition generating non-overlapping subsets, so the relationship $\boldsymbol{C}$ defined in an event space A creates a partition C: $\{A\} \rightarrow \{A_1, A_2, \ldots, A_n\}$. We define t_s as a set of events happening contemporaneously with S: $t_s = \{v \mid v\, \boldsymbol{C}\, S\}$. Events being partly ordered by $\boldsymbol{C}$, it generates a relationship (6). Dealing with the ordered set $\{t_1, t_2, \ldots\}$ we assume the existence of a linear ordering (7).

$$t_s\, \boldsymbol{R}\, t_{s'} \leftrightarrow S\, \boldsymbol{R}\, S' \tag{6}$$

$$\forall t, \forall t' (t\, \boldsymbol{P}\, t') \vee (t'\, \boldsymbol{P}\, t) \vee (t = t') \tag{7}$$

However, the real life events observed and registered by computer sensors not necessarily give an ordered event sequence, especially by $\boldsymbol{C}$. The event collation using a linear ordering in domain of real numbers Z given by $\{Z, <\}$ permits replacing the set by the relation $P\ \{T, \boldsymbol{R}\}$ because of the isomorphism between systems $\{T, \boldsymbol{R}\}$ and $\{Z, <\}$. Usually people use real numbers but computer operates on discrete scale. Now we can date events and discuss about the linkage between $(S_1\ \boldsymbol{R}\ S_2)$ and $(S_3\ \boldsymbol{R}\ S_4)$. We note the lack of the first and the last elements, reflecting the fact that the time has no start and no end, and the homogeneity of the points, because in real life the time scale has no singularities.

Real life events fall into two categories: events which can be repeated again or those happening once. Rickert and Windelband speak about the nomothetic (repetitive) events and idiographic ones [18]. Switching light, washing hands, opening doors are nomothetic, while the patient death instance is idiographic.

In real life the events manifested become the past. Inside a computer system, an annotated event might persist in software. The event flow becomes digital data, enabling the interpretation and the pre-programmed reaction. Assuming $\{A_1, B_1, C_1\}$ an originating pattern, the computer might release some feedback, which might happen after or even before the human actor commits the next $\{A_2, B_2, C_2\}$ chain. It is possible to capture and process an "error" maturing along the decision tree. The time scale describing objective characteristics of the real world exists independently and can be observed becoming an objective. Every "last", "present", or "current" event becomes the past, being replaced by new events, but the desired "next" event can be calculated, being known and formalised a priori the "correct" relationship $\boldsymbol{P}$ – hand washing, mask wearing, and gloves use before entering in the patient room – along the event chain. Desired future event is something being calculated but not manifested yet, while an unwanted event violates "our" rules. The computational model for certain local phenomena might be encoded, and should the computational power be higher than the natural maturity rate, we obtain early warning anticipating the event to be manifested by Nature.

In real life the new present happens only after cleaning the space needed to manifest the phenomena: it might happen only after the previous one is moved in the past, becoming non-accessible anymore. The computer would (a) compute the new event assumed to be materialised as desired future/next, (b) compute the placeholder for the new past, (c) move the previous present in the past using the placeholder created, and (d) materialise the result as the new present event. The calculation of the future and the past is reflected by the cause–effect relationship.

Any possible near miss in BMT nursing we aim to reason about is a *future event*, an event to be manifested, while any judgement is possible over the past events. Starting from an ordered set of events M, where M_i is the present $\mathbf{h}$, we have to process/clean some M_j, where $j<i$, obtaining new M' keeping the memory/trace of M and becoming the new $\mathbf{h}'$ (8), e.g. following steps forming a closed loop.

$$H'\ \boldsymbol{R}\ h\ \leftarrow \exists\, t(h'\ \boldsymbol{R}\ t)\ \text{AND}\ (t\ \boldsymbol{R}\ h) \tag{8}$$

Consequently we consider the time as the discrete computational process/sequence populated by discrete elementary operations, repetitive or not, happening in the virtual digital word named Internet of Things. Our time reasoning starts from the incomplete quantity of information available at t_0 – information growing along the timeline – and the decision is based on the steadily raising error's probability along the decision tree.

7.2.7 Behaviour Formalisation

The human actor behaviour can be described by its *objective function* correlated with the goal's achievements. The set of resources that the human actor uses includes money, energy, vital resources, time (shift duration), information, and knowledge. The determination to reach the above-mentioned goals becomes the motive and the sense, while the degree of the goal achievements, reflected by the objective function, determines the personal satisfaction.

Let us denote with i the index of the goal, m – the number of the goals, γ_i – the importance of the goal i, Target_i – the planned achievement level of the goal i, Fact_i – the real achievement of the goal i, x_i – the amount of material resources (money, food, drugs, etc.), y_i – the amount of energy (real and/or vital), z_i – the amount of time available, g_i – the amount of the information/knowledge available to materialize the goal i, Ω_i – the quality of the institution – the actor is working in – needed to achieve the goal i, while φ and ψ represent the functions linking the vital resources with obtained results and describing the product/service. We assume an actor owns initially the following asset: $\{x_0, y_0, z_0, g_0\}$. The statement (9) defines an Objective function, rewritten as (10) to simplify it. Statements (11) and (12) represent the functional limitations, while (13) describes the resource limitations.

$$\Theta(t) = \sum_{i=1,m} \gamma_i(t) \left(\text{Fact}_i(t)/\text{Target}_i(t)\right)/m \rightarrow \text{Max} \tag{9}$$

$$\Theta(t) = \sum_{i=1,m} \gamma_i |\text{Fact}_i(t) - \text{Target}_i(t)|/m \rightarrow \text{Min} \tag{10}$$

$$\text{Target}_i(t) = \varphi(t, x_i, y_i, z_i, g_i, \Omega_i),\ i \in [1,m] \tag{11}$$

$$\text{Fact}_i(t) = \psi(t, x_i, y_i, z_i, g_i, \Omega_i),\ i \in [1,m] \tag{12}$$

$$\sum_{i=1,m} x_i(t) < x_0(t),\ \sum_{i=1,m} y_i(t) < y_0(t),\ \sum_{i=1,m} z_i(t) < z_0(t),\ \sum_{i=1,m} g_i(t) < g_0(t) \tag{13}$$

According to the above-mentioned behavioural model the vital resources are consumed to achieve the goals. The activities undertaken by human beings fit the model while resource limitations should reflect the consumption dynamics. Time available is constant: it might be 24 h or 8 h corresponding to the shift duration. Vital energy is a renewable resource; however, the diminishing trend is accompanied also by the

variable latency. Knowledge is also a renewable human capital, which might grow or decrease. However, the formalised knowledge reflected in the computerised system can be assumed constant. The component of the nurse behaviour conditioned by the structure is also reflected by $\Theta(t)$.

7.3 Technologies for Near Miss in Nursing

To detect near miss we need several technologies to identify stakeholders, patients and nurses, objects (things), and actions. Semantics play the enabling role of the machine reasoning about happening events, allowing comparison between the real life situations and the modelled ones. The software tool considers *patterns captured* by event monitoring component and processes them against formalised rules contained in the knowledge base. We need the formalised knowledge about the domain, identification technologies, event handling, context awareness, pattern matching, machine reasoning, and some complementary technologies.

7.3.1 Identification

Radio frequency identification (*RFID* hereafter) is a common name for *identifying objects* by radio frequency waves, which refers to short-range communication systems consisting of transponders or tags and scanners, transceivers, or readers. RFID tag is usually a small and cheap electronic circuit with an antenna tuned to a particular frequency. The scanner can detect and communicate with the wireless tag. RFID systems automatically identify objects and gather information about systems, and such applications are widely used: cards opening doors, public transport tickets, ski lift passes, electronic car immobilisers. Asset tracking is an obvious use, where RFID transponders are placed within a network of sensors tracking items through the whole supply chain. Access control systems are a kind of personal identification. RFID term is used to denote various technologies that use radio waves to automatically identify various things, such as people and physical or non-physical objects. RFID technology is similar to – and it inherits from – the bar code identification systems widely adopted. However, RFID does rely on the wireless communication loop, instead of the optical one, exploiting the electromagnetic coupling in the radio frequency spectrum to transmit signals.

Any RFID system includes two main components: a “reader” and a “tag”. A smart electronic component stores the information. An embedded antenna, being activated, does transfer data to a processing device, which calls the application layer to generate an event accompanied by some knowledge. An important point is the fact that the RFID operation provides the means to pick up the information from any object and convert it – using the radio waves reflected – into digital information. The above-mentioned digitisation maps directly into the Internet of Things world, enabling computers to analyse the data and the knowledge. There are two families of

RFID systems, based on two different types of tags used: the cheapest passive tags are those scavenging some energy from the reader to transmit their signal, and the more expensive active tags, those having on-board battery to power the data signal transmission over a distance. Let us analyse a typical RFID reader (Fig. 7.3) from a new viewpoint: RFID reader interacts with the tag and sends a signal to someone for further elaboration replacing the interaction by an event enriched by some *context* information.

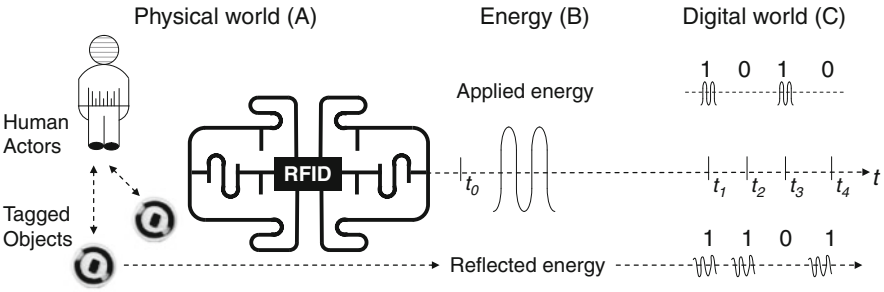


Fig. 7.3 RFID transformation

RFID-enabled system [19] interprets the data and takes some decisions. RFID tag is a physical object, applied to a real-world item in order to identify and track it electronically. The tag makes the physical object an energy consumer, because RFID technology uses energy to activate the sending of data. The quantity of energy scavenged might be very small, but any item – previously energy-less – becomes an energy consumer. Tagged physical objects are being introduced into the digital world, and the real life behavioural characteristics are inherited and represented by assigning to such digital objects *semantics*. RFID and knowledge technologies become the transforming function able to map physical objects into the digital world by the simple injection. Several implications are generated by this very simple assertion: we obtain the digital representation of a real-world object: its digital clone. We see (Fig. 7.3) the energy-less physical world member identified by tag (*A*), some analogue energy going from the real-world power plants and some reflected energy (*B*), and the digital signal (*C*), ready to be consumed by the digital world. We propose the above-said transformation to project physical objects vs. the Internet of Things. A real-world object or living organism (*A*) made identifiable by the tag (*B*), becomes a digital object of the Internet of Things (*C*) and a consumer of the energy business: a digital Thing hereafter. Domain stakeholders – nurses, patients, doctors, and so on – and items used in the everyday practice – gloves, masks, drugs, etc. – send own characteristics accompanied by some additional context-related knowledge about the time, place, and external events to the knowledge system. Digital object *C* becomes an entity capable to originate a *flow of events*, enabling the decision-making, process control, and further triggering options (14). The computational methods owned by class **C** are instruments used to actuate programmed reactions (15), while events originated by class **C** broadcast the ID of the actor, the semantic meaning of the action, the time, the place, and hopefully the context (16).

$$\mathrm{C} = \mathrm{RFID\_Transformation}\ (\text{real object } A, \text{ tag } B, \text{ time, place, context}) \quad (14)$$

$$\mathrm{Reaction}\ (\mathrm{Event}_j) = \mathrm{C} \rightarrow \mathrm{Method}_i \quad (15)$$

$$\mathrm{Event}_j\ ([\mathrm{Event}_1, \mathrm{Event}_2, \ldots, \mathrm{Event}_j, \ldots], \mathrm{ID\_actor_j}, \text{time, place, context}) \quad (16)$$

There are also new possibilities to control directional processes and procedures, offered by knowledge technology. Every geographically distributed process can be split and decomposed into several businesses (Fig. 7.4): an industry producing goods (A), a transport business (B), a value-added service industry (C), a nested small business (D), a possible subcontracting (E), a distribution/delivery chain (F), and any external business (G). We map processes into the digital world organising the interoperable business, transforming any process, which is a non-physical object, into an energy consumer. This transformation maps non-physical objects into digital things as well.

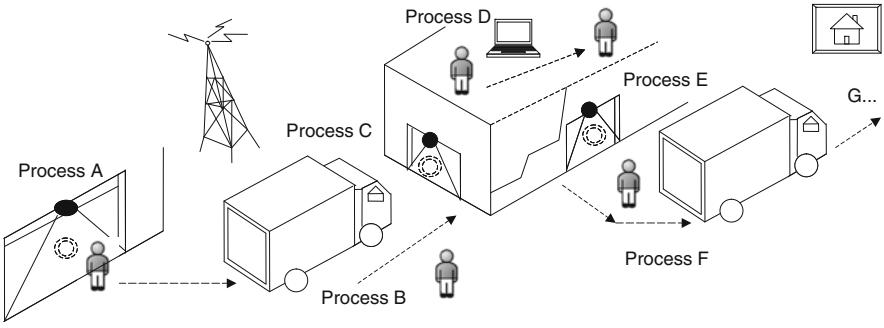


Fig. 7.4 Process mapping

There are different interaction forms: human interacting with the real-world objects, human interacting with the digital world objects using PC or something else, human interacting with the real-world process, human interacting with the digitised process, etc. Radio science and technology innovate [20] interactions between PC and similar devices used to manage the real world and the ubiquitous digital one around. In BMT nursing the main actors are nurses and patients, while the main processes are everyday operations like going inside the patient room to assist him and so on. Giving to a patient one RFID bracelet and forcing nurses to use the personalised RFID readers, we obtain two identified objects capable to show the interactions among them, e.g. digitising the human-to-human interactions.

Several examples in literature describe how the RFID applications are used in healthcare industry, ranging from the patient identification using RFID artefacts, drug boxes identification using RFID sticks, to more sophisticated applications comparing the drug prescription with the drug administration; however, almost all described classes treat the human-to-thing (machine) cases. In other domains, such

as commerce, logistics, transport, and similar, the application classes describe the machine-to-machine scenario, lacking the most challenging case, namely human-to-human interaction. RFID technology has recently enabled remote identification, cheap real-time tracking of objects, and high-speed communication over short distances. This gives the possibility of acquiring detailed information about real-world systems, to trace patients and material in hospitals in order to optimise the healthcare process. It can also be used in smart devices to minimise human errors: a system double-checking drugs administered to a patient in order to avoid mistakes. We use RFID technology to capture events and identify interactions. We send digitised interactions to the knowledge layer for processing and decision-making in order to obtain the reaction in real time. Several sensors are available for telemonitoring services. By combining RFID cell phones and RFID sensors with cellular networks or the Internet, the stakeholder (patient, doctor, nurse) becomes empowered to read RFID sensor tags anywhere for almost any application. RFID reader interacts with the tag and sends a signal anywhere to someone for further elaboration. Recent examples include glucose monitoring, cardiac monitoring, UV monitoring, and biomarker skin test patches. RFID-enabled system interprets the data about the physical objects and takes decisions locally or remotely, after communicating digital objects. Processes might become digital objects too: a system composed of two RFID readers is capable to discriminate directions. Real world includes physical and non-physical objects.

In outdoor applications the RFID readers might be synchronised using GPS technology; however, the in-hospital real-time systems should consider different time-sampling techniques to produce the events referred to time.

7.3.2 Ubiquity

Digital Hospital might be seen as the free *flow of real-time information* exchanged between healthcare professionals and patients: digital need of care, digital multimedia objects characterising the human being and related processes, digital care service, and digital money. The process of healthcare digitisation starts exploiting the always-on broadband, mainly Internet connections, public–private synergies, the rise of merchant-assistive services, and ubiquitous trading of services. The Digital Hospital, like any abstraction, falls into the Internet of Things (*IOT* hereafter) sphere because of similar implications. We can ubiquitously manage on-line bank accounts, digital meters, industrial automation equipment, intelligent home, so it is timely to speak also about the telemedicine and assistive services. Digital Hospital will be the abstraction working in background fully automatically, which will be invisible to the user, but such system-of-systems will be able to manage the digital multimedia objects worldwide.

IOT is a new vision of the technological ubiquity in communication era radically transforming the society, corporate, communities, and personal spheres. Early forms were represented by the widespread use of mobile phones, little gadgets becoming an integral part of everyday life. Second step is the embedding of short-range mobile

transceivers into a wide array of additional devices/appliances, enabling new forms of communication between People and Things (P2T) and between Things (T2T). A new dimension has been added to the world of information and communication technologies: from anytime, anywhere connectivity for anyone, to anything, e.g. between PCs, Human to Human (H2H), Human to Thing (H2T), and Thing to Thing (T2T). Multiple connections create an entirely new dynamic network of networks – an Internet of Things, based on solid technological advances and visions of network ubiquity, computing, communications, and dynamic technical innovation in a number of domains, ranging from wireless sensors to nanotechnology. Embedded intelligence, the advances in miniaturisation and nanotechnology, and a combination of all the above-mentioned developments make the IOT. It connects the world's objects in both a sensory and an intelligent manner mapping them into the digital representation of physical objects.

An enhanced message-oriented RFID middleware, filling the gap between the Internet of Things and Internet of Services is capable to preserve semantics, elicit the knowledge from the data, and deliver the events characterising fully what is happening on the remote node. All the time labelled events might be processed locally or remotely; however, the time synchronisation is required for the distributed knowledge elicitation.

7.3.3 Adaptivity

Personalised healthcare is a challenge because it supports the sustainability of care. Internet of Things is a paradigm promising to manage the *digital identity*, so the personalisation of care services. Different equipments are used in extra-wall healthcare and assistive services, requiring different sorts of objects to communicate and to make the ubiquitous system-of-system. Extended entities and mixed roles are becoming interoperable. We present how adaptative care might exploit the available ICT, knowledge, and broadband technologies.

Personalised healthcare might be seen as an abstraction layer of IOT. Documents (laboratory exams, X-rays) represent the lowest one. Services, such as medical records, X-ray imaging, blood test results management, medical applications, represent the middle abstraction layer, while a disease treatment, e.g. "anaemia treatment", illustrates the higher abstraction layer. A collection of highest possible layers becomes the Personalised Healthcare System (*PHS* hereafter) entity of IOT. Modern PHS systems use Internet connection to link the remote node (incl. home in e-Health) and the caregiver. The reliability of the link should be considered because of the nature of the care service. Any telemedical equipment is alimented by electric power; so twisted pair wired setup is an asset to exploit. The powerline communication and fieldbus-enabled systems transforms the electrical energy distributions networks into the valuable components of the IOT, where the embedded chips and RFID items grant the identification capability: Digital Hospital depends on reliable communications. Wireless sensors gather real-world data; body and ambient sensor network services generate both behavioural and biological data. Unobtrusive service

delivery through computers and consumer electronic devices with various interfaces distributed throughout the home, and proactive human-machine interfaces (HMI) with context-aware sensing, motivate health-conscious behaviour.

The interoperable business deals with both of them. The digital things in the digital world are collections of objects to exchange and to make them interoperable. To interact with a process, the actor (human) should use a tag, which might be embedded into a mobile phone, wristwatch, or plastic badge. All different interaction forms, human interacting with the real-world objects, human interacting with the digital world-objects, human interacting with the real-world process, human interacting with the digitised process, might be represented by digital items. An object becomes a data element, while a process becomes a collection of data elements with appropriate timing. To discover the direction of the movement, the timestamp is analysed. Hence the synchronisation of clocks among the RFID network is important. The synchronisation is the ubiquitous service as well, and it might be performed over the Internet or over the fieldbus topology. The final vision of the Internet of Things includes enabling technologies such as RFID and knowledge/semantics and the traditional "old" world with industries, extended and de-localised enterprises, ubiquitous workers, homes, energy utilities, TLC, services, and all other realities.

The extension of the above-mentioned reality by new enabling and emergent technologies is represented by the knowledge component, including the formalisation of cognitive processes, machine reasoning, Semantic Web, automated decision support, and whatsoever. This is the wireless extension worldwide enabling the future human-less scenario, in which the automated intelligence might govern things: physical objects and non-physical entities (processes), parts of the real world or of the digital one. The intermediate layer shows the missing element, making possible such scenario, which might be the RFID transforming real life members into digital things of Internet of Things.

7.3.4 Sensing and Multi-sensor Fusion

Real life events captured by sensors become an observable object bringing the cause–effect relationship to understand or govern. The *event flow* in nursing domain might be exemplified by (17–20).

$$\text{Event (PickUpGloves, ID\_Actor1}, t_1, \text{ID\_Item1, } \ldots) \tag{17}$$

$$\text{Event (WearMask, ID\_Actor1}, t_2, \text{ID\_Item1}, \ldots) \tag{18}$$

$$\text{Event (OpenDoor, ID\_Actor1}, t_3, \text{ID\_Item1}, \ldots) \tag{19}$$

$$\text{Event (DispenseDrug, ID\_Actor1}, t_4, \text{ID\_Item1, ID\_Actor2}, \ldots) \tag{20}$$

We can detect domain events happened using different techniques and sensorial devices. The identification technology suggested is the RFID because it facilitates the convergence with the Internet of Things. The presence could be captured accounting the interactions with material things, like labelled drugs, gloves, masks, or chairs and whatsoever. The mobility – or directional processes – might be perceived and characterised using synchronised couples of proximity sensors or RFID: two events happening at t_1 and t_2 moments could be considered as a directional vector with a timing label. Single sensor approach is useful to deal with a well-known and formalised phenomenon. Multi-sensor fusion approaching holistically multiple information sources relies on sensors reducing both systematic and random errors. For instance, in cardiac sensing both ECG and haemodynamic signals, such as the impedance cardiograph or blood pressure, give mutually correlated information, being physiologically coupled mechanical and electrical functions. Whenever an ECG signal is degraded either due to poor electrode connection or patient movement, joint analyses of complementary sensors, such as the ventricular pressure, sustain cardiac monitoring resolving some intrinsic ambiguities in rhythm disturbance assessed by ECG alone. While error minimisation by multiple identical sensors appears intuitive, the reliance on different sensors in terms of both sensing type and location exploits general principles of pattern recognition and machine learning. Further challenge is the redundancy introduced by the adoption of the powerline communication together with the traditional one (medical equipment is alimented by energy) exploiting the multiple parallel communication channels offering the better resilience.

From another point of view, two different classes of sensors permit detecting the same event in the time space at t_1 and t_2 moments or the time arrow, corresponding to the early detection (t_1) and the usual perception (t_2) of the phenomena. This enables the formulation of the hypothesis about a happening event through the measurable pattern, generating the situational knowledge leading to the decision, potentially before t_2. While error minimisation by multiple identical sensors appears intuitive, the reliance on different sensors in terms of both sensing type and location exploits general principles of pattern recognition and machine learning. Going beyond, the machine-reasoning system can receive in input the pattern characterising an event happening in time space at t_1. The time interval needed to decide about the possible evolution of the event might be important in certain situations, especially in intensive care units, when the real-time reaction would be useful; however, the true power stays in the fact to generate the forecast about the event tree and to assess the next event at t_1' against the rules characterising the normality or abnormality, e.g. about an error or a correctness of the intended action to undertake. The possibility to react automatically before the t_2 – or before the t_2' – leads to the near-miss management.

As a simpler example, let us remember that we can detect a hand movement event using mechanical device (mouse), optical device (video camera), ultrasonic sensor, Terahertz device, etc. Considering the different latency, an observation of the same event – happened at t_0 – will generate the series of annotated events timed t_1, t_2, t_3, etc. An interesting viewpoint is the fact that at t_1 the observed cause of

the event is already in the past – which is t_0 – however, the computerised system is aware about the event in the present time t_1, but the qualifications can certainly arrive in the future at t_2, t_3, t_4, etc., where t_4 might be the error-committing point, but t_3 might be sufficient to capture a near miss, because of the natural latency of the human being. It is clear that the degree of the certainty E might be different at t_1, t_2, t_3 and t_4, being $E(t_1) < E(t_2) < E(t_3)$; however, the Fuzzy reasoning becomes an efficient instrument to manage the problem. The time synchronisation in distributed systems makes possible the appropriate ordering of the time-labelled entities.

7.3.5 Presence and Context

Proximity (infrared) sensors can capture the use of the surgical scrub sink by someone and generate the information about it. The combination of two sensors, proximity and RFID ones, interacting with the nurse wearing the RFID identifying bracelet, gives the information about the person using the sink and how long it takes to wash hands. The tagged gloves, surgical masks, boxed drugs, or Chlorhexidine bottles interact similarly with the human actor representative device, saying who is operating in proximity of the box, containing the above-said items.

The event flow is characterising *indirectly* the said interactions, because it gives no guarantee that the human actor has effectively withdrawn an item from the box and has really used the tagged object we are speaking about. The event says simply that an interaction between two objects is captured, while the context complements the facts. In theory, we can use the video camera filming the interaction (process) and apply the image processing to the sequence of snapshots classifying the operation, but it is a complex task offering a poor cost/benefit ratio. Local intelligent devices become capable to originate an event flow about real life happenings, sufficient to set up a computerised system accounting real life situations and reacting to them. An interesting overview about the presence in smart homes can be found in [21, 22].

There are several other enabling wireless technologies helping to handle the human-to-human interactions and the presence (co-presence), among which we refer ultrawide band [23] accompanied by the use of various skin patches and Terahertz ones [24], offering alternative ways to digitise the human-to-human interactions using electromagnetic induction coils or other artefacts for tagging purposes, reducing invasiveness, but increasing costs and complexity.

Ultrawide band is a low-energy radio technology for short-range high-bandwidth communications using pulse-coded information. Most recent applications target sensor data collection, precision locating and tracking applications, improving the context's quality. The advantage is the low invasiveness and the high precision, while the drawback is the high complexity in context recognition.

Terahertz radiation is non-ionising, not damaging tissues and DNA. Terahertz radiation can penetrate fibres, plastics, and several millimetres of tissue, enabling imaging, surveillance, security screening, and so on contactless and remotely.

This new technology is promising, but actually expensive and immature: real life application classes in the life science domain are still to appear.

The less known work is contained in A. Bobrov's monograph "*Model Research of Field Consciousness Mechanism*" [25], where the author describes the use of biologic and physical detectors relying on the electrical double layer phenomena to reveal the human-to-human interactions in a contactless way. He described the simplest physical detector (Fig. 7.5) – a glass container filled by some water with two electrodes, one of which is touching the surface – capable to detect the human presence in a given place remotely. Considering the main drawback – a significant latency in recovering the initial state after the interaction (Fig. 7.6) – the phenomena described by Bobrov would be useless in real-time applications. However, it enables the forensic a posteriori assessment of the caregiver's presence in the patient room and can contribute to characterise the context, so acquires the better context sensitiveness.

The above-mentioned issues are relevant to acquire the context awareness and so to pass to process the event flow.

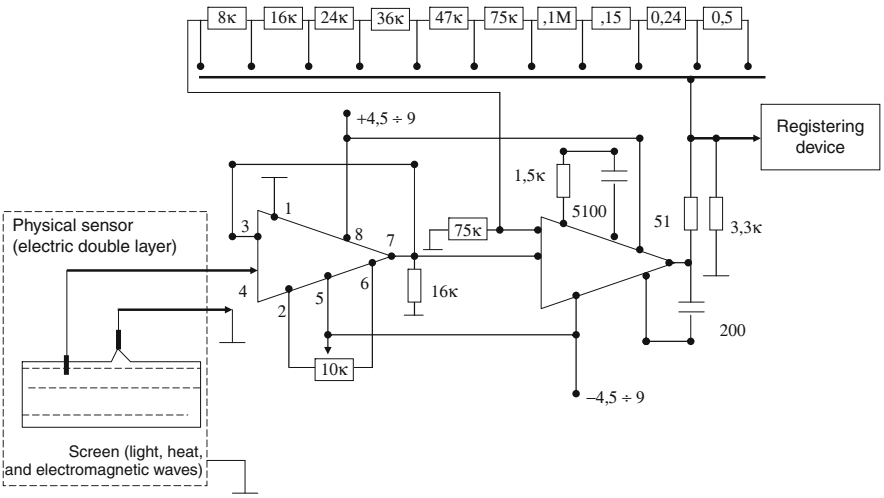


Fig. 7.5 Bobrov presence sensor (courtesy by author)

7.3.6 Uncertainty and Rule-Based Systems

Fuzzy sets are sets whose elements have degrees of membership. Fuzzy sets have been introduced by Prof. Zadeh [26] as an extension of the classical notion of set. Fuzzy set theory permits the gradual assessment of the membership of elements in a set; this is described with the aid of a membership function valued in the real unit interval [0, 1]. Fuzzy sets generalise classical sets, since the indicator functions of classical sets are special cases of the membership functions of fuzzy sets, if the latter

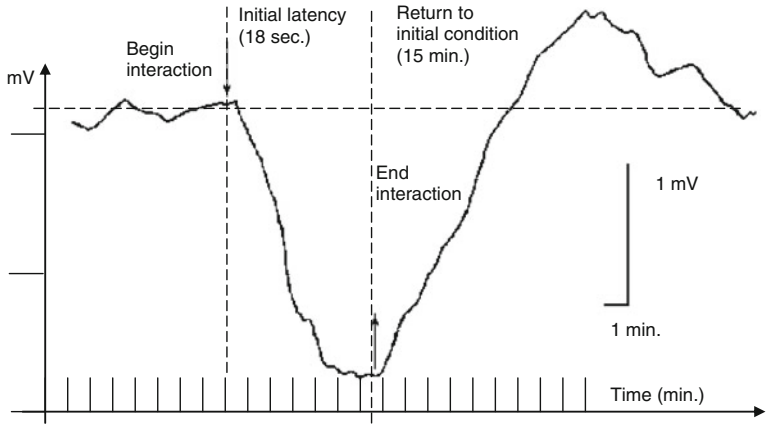


Fig. 7.6 Bobrov presence experiment annotated

only take values 0 or 1. Fuzzy Logic [27, 28] is derived from the Fuzzy set theory dealing with reasoning that is approximate rather than precisely deduced from classical predicate logic. Fuzzy Logic is a way of processing data by allowing partial set membership rather than crisp set ones. People do not require precise, numerical information input, and yet they are capable of highly adaptive control. If feedback controllers could be programmed to accept noisy, imprecise input, they would be much more effective and perhaps easier to implement. Fuzzy Logic might be seen as a problem-solving control system methodology that lends itself to implementation in systems ranging from small to large, networked computerised data acquisition and control systems, which can be implemented in hardware, software, or a combination of both, providing a simple way to arrive at a definite conclusion based upon vague, ambiguous, imprecise, noisy, or missing input information.

Fuzzy Logic incorporates a simple, rule-based approach to solving a control problem rather than attempting to model a system mathematically. The model is empirically based, relying on an operator's experience rather than their technical understanding of the system. For example, rather than dealing with terms such as "10 s. < Time elapsed < 20 s.", or "1 atm. < Pressure < 1.1 atm.", imprecise terms like "IF (Time is sufficient) AND (Pressure is leveraged) THEN (Process is normal)" are used, being very descriptive and meaningful of what must actually happen. Fuzzy Logic requires some numerical parameters in order to operate such as what is considered significant error and significant rate-of-change-of-error, but exact values of these numbers are usually not critical unless very responsive performance is required in which case empirical tuning would determine them.

Fuzzy rules are linguistic IF–THEN constructions that have the general form "IF A THEN B" where A (*premise*) and B (*consequence*) are collections of propositions containing linguistic variables. In effect, the use of linguistic variables and Fuzzy IF–THEN rules exploits the tolerance for imprecision and uncertainty. In this respect, Fuzzy Logic mimics the crucial ability of the human mind to summarise

data and focus on decision-relevant information. In a more explicit form, if there are i rules each with k premises in a system, the ith rule has the following form (21):

$$\textbf{IF}\,(a_1\,\textbf{IS}\,A_{i,1})\,\Theta\,(a_2\ \textbf{IS}\ A_{i,2})\,\Theta\,\cdots\,\Theta\,(a_k\,\textbf{IS}\,A_{i,k})\,\textbf{THEN}\,B_i \tag{21}$$

where $\boldsymbol{a}$ represents the crisp inputs to the rule and $\boldsymbol{A}$ and $\boldsymbol{B}$ are linguistic variables, while Θ is a Boolean operator AND, OR/XOR, or NOT. Example: **IF** (the time to wash hands **IS** insufficient) **THEN** (the system warning **IS** released). Several rules constitute a Fuzzy rule-based system.

The human being has the ability to discard most of the information and to concentrate only on the information that is task relevant, achieving a capability to process Fuzzy information. Filtering task-relevant information in the information over-flood is reduced to a manageable level. We formalise notions like "appropriately washed hands" formulating algorithmically processable rules using soft computing.

Uncertainty is a fundamental and unavoidable feature of real life. In order to deal with uncertainty intelligently, we need to be able to represent it and reason about it. Further reading is available in [29]. Fuzzy Logic systems address the imprecision of the input and output variables by defining Fuzzy numbers and Fuzzy sets that can be expressed in linguistic variables, e.g. "small", "medium", and "large". Many of the existing systems need the rules to be formulated by an expert.

A knowledge representation formalism proposed in [42] and adopted by authors is based on the combination of three concepts: (a) frame-based knowledge representation formalisms, (b) unification, as found in unification-based grammar formalisms, and (c) fuzzy set theory. The above-mentioned formalism has been developed for and used in the technological framework for knowledge management, whose main feature was the ability to semantically index documents by representing the (uncertain) knowledge about their content and relating it to ontology. That framework was validated by means of three vertical applications.

7.3.7 Ontology Engineering in Bone Marrow Transplantation

To understand phenomena happening in real life we need the knowledge about the domain and the event space, consequently we need an instrument to describe objects, relations, and rules. Ontology has its origin in philosophy, with special attention to the relations between particulars and universals, between intrinsic and extrinsic properties, and between essence and existence. According to Tom Gruber from Stanford University the meaning of ontology in the context of computer science is a description of the concepts and relationships that can exist for an agent or a community of agents, representing entities, ideas, events, their properties, and relations, using a system of categories to reason about the properties of the domain or to define it. What ontology has in common in both computer science and philosophy is the representation of entities, ideas, and events, along with their properties, and relations, according to a system of categories. It is used to reason about the properties of the domain and may be used to define it. We operate in the life science domain

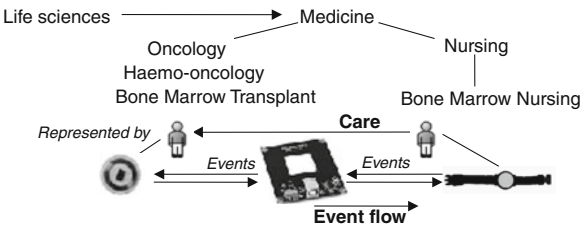


Fig. 7.7 Interacting objects

where the traditional medicine is one of the top-level structures we start from. The bone marrow transplantation subject area can be seen as a sub-domain of oncology, belonging to the domain of medicine (Fig. 7.7).

The central concept is the bone marrow with its pathologies and the *care processes and operations*, including the transplantation. In order to reason about the nursing events we have to model all interactions which are allowed, relevant, and notably correct. A possible approach for ontology engineering permitting to describe the bone marrow transplantation domain is published in [30] where authors illustrate the ontology construction exploring T-cell depletion, a special issue in BMT, aiming to support the text summarisation. Chris Catton from Oxford University has published an experience in building an ontology-driven image database for biologists, featuring a section of an ontology describing the development of adult mammalian bone marrow and brain. Some details about the BMT can be found in Gene Ontology (online resource on www.geneontology.org).

Some aspects of the transplantation process can be found in [31]. BMT nursing is a sub-domain of the nursing, focusing primarily on the care processes and operations, modelling actors such as nurses and patients and so on. In response to the need to support information requirements, nursing has developed a number of different terminology systems. In [32] authors show that the most common approach in ontological engineering is not optimal to enable the machine-processability of the nursing operations. Nick Hardiker has been dealing with nursing ontologies [33] for a number of years describing the mediating interventions [34] and ontology [35]. However, the development of formal nursing terminology systems is not an end in itself. To operate with care procedures we should consider the use of the process ontologies and the scope of the concrete real life application to instantiate. The major obstacle towards a view on nursing processes is that the elementary processes are widely not accessible to machine reasoning.

Paik [36] operates with *nursing care plans*, notably the source of knowledge involved in the overall nursing process, which leads up to the nursing diagnosis. Being known that the nursing care plans are the embodiment of *how* nurses apply a clinical reasoning process to analyse and evaluate gathered facts about patients and real life situations, it appears justified to apply the description and situation pattern [44] to solve some issues from this problem area.

Nurses express their knowledge about care processes in terms of Fuzzy rules, and guidelines are translated into formalised rules too, prior to migrating into the

knowledge-oriented computerised system. Our process conceptualisation specifies the behavioural view on care. BMT domain, workflows, processes, and guidelines are represented by ontologies, while the worker behaviour and the activity generates a flow of facts (17–20) such as "$Nurse_1$ is entering the anteroom A_1 at 6.34", "$Nurse_1$ is washing hands at 6.35", "$Nurse_1$ is opening the $Door_2$ at 6.36". The match between the rule "first wear the mask and use gloves then", and the situation "mask and gloves not used prior to 6.36" is done by computerised brain.

Near-miss detection in BMT is vital, and real-time proactive help might be ensured by automated computerised decision support enabled by machine reasoning, requiring some more components: the representation of the domain and situations, events, phenomena to control, e.g. the ontology containing the universal knowledge and the knowledge repository, which is a container of instances. We research on semantics in BMT nursing process analysis for the evaluation of the current running processes. In our approach the nursing actions/processes are represented as concepts with the RDF payload holding the data elements enabling the process assessment. The instances are generated by RFID equipment digitising the human/nursing actions and their values reflect the individual nursing behaviour. We elicit the knowledge elements characterising the nursing actions and process them reasoning about the modelled error patterns with the aim to find potential near miss and interrupt the respective "unhappy" event chain.

7.4 Knowledge and Semantics

Healthcare is one of the most information-driven industries: primary data collected from a series of discrete observations bear imminent information but overload, so the knowledge is typically derived by systematic interpretation of the essential structured information, exploiting classification models in the medical field. Medical history, nurse notes, description of pathways, protocols of care (information about treatments and diagnosis) are unstructured multimedia; a proper classification of the underlying concepts is needed. Natural Language Processing and ontology-driven clinical DSS mine facts on a particular patient, and guidelines and protocols' triggering enable classification, because a systematised taxonomy of medical terminology and concepts identify instances in available corpus. Another option is represented by Fuzzy systems trying to understand events happening in the patient room, classifying event chains against the normality.

Diagnostics look for the person's condition change between examinations, rather than the current snapshot at the time being status, requiring evolving pattern detection in dynamics. Some parallelism with the representation of digitised processes is envisaged. A current mammogram has clinical value, but its true significance stays in *relation to the past* one: two objects should be recovered, transmitted, elaborated, and compared. Something that appears suspicious but unchanged is less relevant than the situation evolved radically in the last 6 months. Since any disease is an evolutionary process, the data-warehousing techniques and the time reasoning

should be considered. However, the previous mammogram is frequently unavailable because of other facility visit and interoperability lack: each of us should leave a distributed multimedia library behind allowing PHS to bring all the information together to the point of need on time. Healthcare adopts also uncertainty reasoning and needs the context-related information (Fig. 7.8).

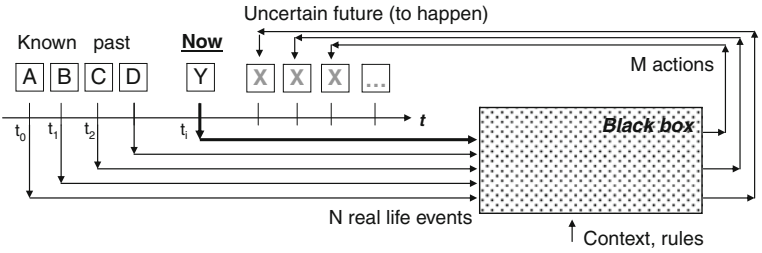


Fig. 7.8 Semantic mining as the "black box" system

BMT nursing is a complex process decomposable in the chains of the possible actions. We have observed some specificity of BMT nursing processes, because the events manifest some recurrent patterns, detectable by sensors. The conceptualisation of the above-mentioned patterns in the ontology makes possible the automated pattern matching. In our approach we link the event's data with defined ontological concepts. This linkage provides concept-based knowledge elements enabling automatic processing and machine reasoning. In our approach a semantic analysis has three steps: (a) the creation of an ontology capturing the meanings of tasks, data elements, and performers in processes; (b) the semantic annotation of BMT nursing processes with the defined ontology; and (c) the definition of semantic constraints triggering the events/happenings. The data mapping within an ontology framework comprises the relevant concepts for events description and process understanding.

After the "ideal" nursing processes being annotated using the ontology, by mapping them at design time onto the processes, tasks and sensorial data elements accompanying such processes should be made coherent with the conceptualisation adopted. It is done using the architecture and hardware components monitoring BMT nursing processes at execution time. In this way we ensure that the run-time event logs will contain the necessary references to those ontology concepts, ensuring the possibility to trigger the events in real time. The tool mainly focuses on pattern discovery and scenario-based analysis using predefined models or Description and Situation templates [41–46] discriminating between the normal and abnormal situations. We deal with a possible extension in semantic process discovery (auditing) and semantic performance analysis. Using hierarchical models based on subsumption trees of the event chains, the semantic discovery automatically detects patterns based on task similarities and overcomes the current process mining limitations, capturing a flat/static representation of process models. The nursing events captured by sensors in real time are made available as event sequences to the knowledge engine, applying the above-mentioned semantic techniques.

At the present time the current techniques lack the capability to deal with the notion of the "execution time" and the time sequences. BMT nursing process components are happening in the time dimension of the 4D time space, and the correct timing of atomic processes and their composition is the essential part of the problem. Thus the main reason to use and the main benefit coming from the semantic technologies in BMT nursing domain is the challenge to understand the "acceptable" execution time of tasks/processes, since the defined ontology and the fuzzy rule-based system will capture these notions and process them.

7.4.1 Nursing Process Modelling

BMT nursing is a process executed by human actors in real time. In clinical nursing the semantic meaning of care actions undertaken is more important than the semantic payload of documents which is typical for clinical decision support and the corpus categorisation systems. The nursing activities are assembled in different nursing processes happening in time dimension. Each nursing process has a main goal; it consumes some resources, it has a typical duration, and it might be judged as correct or wrong. The nursing processes happening in time might manifest the correct sequence, true delays, omissions, and/or premature actions [37]. We have to model the timing, the sequencing, and discriminate between the correct and the wrong timing (Fuzzy category).

A central venous catheter allows the direct and needleless delivery of nutrition, therapies, and other elements into the bloodstream: guidelines specify how to put in place this measure because frequent punctures increase the infection risk. However, the venous line length depends on the organisation implying the pre- and post-infusion timing personalisation. Bone marrow harvesting is governed by medical protocols. The anti-tumoral therapy is a part of some care processes. Intravenous fluids accompanying the chemotherapy are executed in the isolated environment, so each entry in the patient room is accompanied by specific antiseptic measures. The nursing care in this stage is strictly ruled by precise well-formalised guidelines and by specific – but tacit – nursing knowledge. The specificity of the BMT nursing is that each care process is *accompanied by* precise fulfilments ensuring the absence of the pathogens. Nurses enter in the patient room to deliver antibiotics, antiviral drugs, gastro protectors, immunoglobulins, immunosuppressors, chemotherapy, food, and other particular therapies. Nursing assistants contribute with cleaning and some other operations. The environment being controlled, each operation is accompanied by the set of sub-processes ensuring the sterility value chain [4]. Any care operation starts from the hand washing using water or Chlorhexidine before touching objects/things used for care. The objects are tagged entities, while the human actor is identified by RFID bracelet. Because of the prescription or the circumstances nurses start the sequenced and formalised patterns. The timing is used to model the elementary processes, giving the key to understand real life events happening in the patient room. For instance, the antibiotic prescription (amoxicillin) three times per day implies six entries in the patient room to start and stop the infusions lasting

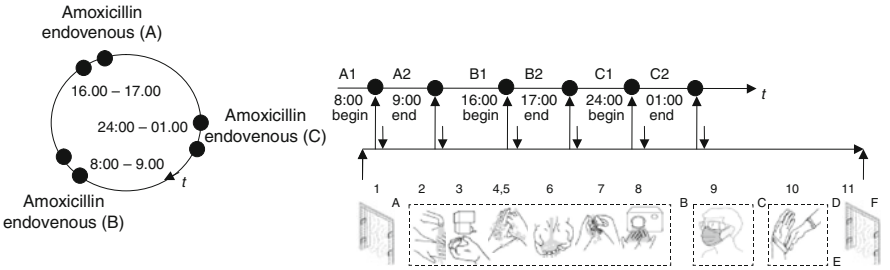


Fig. 7.9 Drug administration process for antibiotics

60 min ca. at 8 and 9, at 16 and 17, at 24 and 1 AM. Each time the accompanying pattern of hand washing, and mask/gloves wearing is manifested (Fig. 7.9).

The end of the infusion might generate some stress should the nurse be busy, because the infusion pump emits sound signal only, lacking the automation interface notifying the end of the infusion. The patient room being separated from the rest of the department by two doors, and calculating the timing of the accompanying processes, we have a real possibility to interrupt an operation generating a warning before the nurse is passing the second door in case a pattern differs from the expected one.

We formalise the key elements of the nursing process transforming the tacit knowledge owned by professionals dealing with the above-mentioned problem into the implicit one, stored into the knowledge repository and controlled by the machine reasoning to detect the near-miss situations.

More complex procedures about antimicotics requiring the pre- and post-operations to ensure the clean venous line are shown in Fig. 7.10. The operation implies three entries in the patient room per event – while the first one lasts 10 min – generating nine entries per day in total.

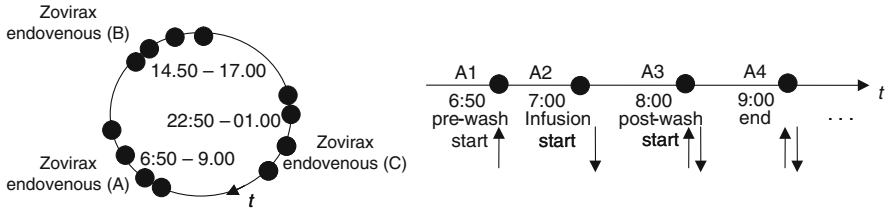


Fig. 7.10 Drug administration process for antimicotics

The most demanding case is shown in Fig. 7.11 because the long-lasting process requires 12 interventions. Moreover the shift lasts 8 h only, adding the entropy because of the changed human actor: the nurse with different identification number might replace the previous one; however, the maturation of the nursing process will continue. The care processes might be composed with overlaps. Finally, generic personnel might execute some operations, like food, nutrition, and room cleaning.

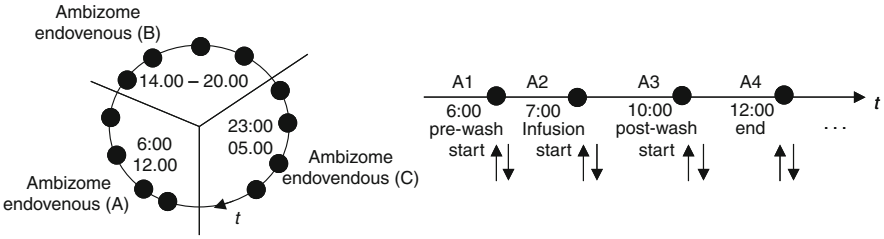


Fig. 7.11 Drug administration process for Ambizome

The modelling aims to describe the correct processes in order to identify the risky situations (to introduce the pathologic agents inside), while the machine reasoning manages and supports healthcare professionals by the proactive help each time the near-miss condition might be envisaged. This help is released before the human actor commits the situation passing through the second door. The original goal is to detect the above-said conditions in real time offering proactive help.

The scientific challenges are (a) human–human interaction to support by automation, (b) directional process-related context understanding, (c) automated reasoning and decision taking. The knowledge support in BMT nursing might be represented by the introduction of some sensor *devices* to detect the risk-characterising *conditions* and the knowledge engine running encoded *rules (descriptions)* aimed on the *situation* triggering to undertake the proactive *action*: to advert the human actor, not acting on behalf. The analysis of nursing practice suggests the introduction of the machine support, offering proactive help especially at the night shift end because of the natural weariness. In such cases the reaction time should appear slowed down, and this phenomenon might be observed and accounted by computerised system. An important BMT infection risk element is the "unsafe" entry in patient room (Fig. 7.12), even if a small, but recurrent element of nursing.

Human factor being constrained, an augmentative computerised proactive personal assistant for nurses able to detect near-miss conditions and releasing a kind of automated support is valuable. We propose the multi-disciplinary mix of advanced knowledge–technology-enriched process control and automated decision support

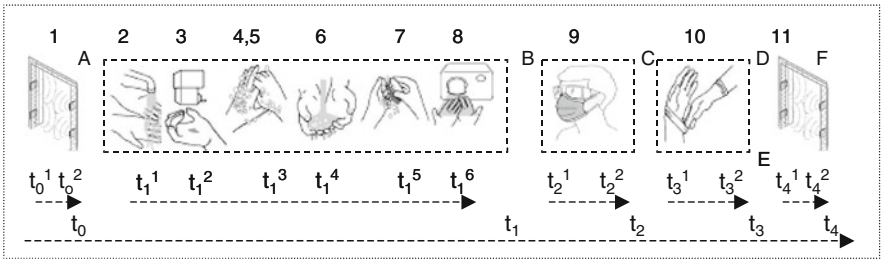


Fig. 7.12 Sequenced pattern

under uncertainty. The complexity of the process is reflected in the richness of technologies proposed, where one component, such as RFID only, would be insufficient to embody a solution.

The "safe" entry – modelled as a plan using Fuzzy terms – is a complex superposition of smaller inter-dependent and constrained elements, such as the entrance in the anteroom (A), mask wearing (B), hand washing (C), gloves wearing (D), air exchange (E), and entrance in the patient room (F). The correctness of the sequence and the timing are important: A → B (A *followed by* B), B → C, C → D, D → E, E → F, both B and E are requiring a "sufficient time", but might run in parallel, while other processes are not, and so on. D prior to C is wrong, but D prior to B is improbable. In several realities F → D and E is continuous accounting time elapsed between two doors opening. We formalise this empiric knowledge in a model to keep in the repository. The hand washing process is not atomic and might be further decomposed in elements marked on Fig. 7.12 by numbers 2–8, to govern by RFID and sensor's application. The in-sink humidity sensor, an algorithm accounting the time elapsed between the opening of the first and second door and comparing it with the time needed to fulfil all prescribed actions, wash hands, use gloves etc., might be proposed. However, in our scope this detail is less important.

The innovation stays in the finding that in BMT nursing almost all processes (like hand washing) are routines, well administered by qualified staff, while the near miss always stays in the *process composition* rather than in components. The wrongly composed superposition of atomic processes generates a near miss if detected, captured and processed on time; becomes an error if detected afterwards; or remains simply unobserved or unobservable. The ontologisation of the routine components gives the semantic meaning and the Fuzzy values characterising the nurses, while the semantic mining discriminates between the normality situations and the error phenotypes. Another innovation is the holistic (H2H) process control under uncertainty assisted by knowledge technology, where our vision goes beyond barcodes and wrist bracelets. Nurse identification is done by RFID, but the fact "IS GOING to enter" rather than "has entered" is detected in real time. The sequence is calculated by IF–THEN crisp rules, and the timing is accounted as a crisp value but the appropriate timing is modelled by Fuzzy trapezoids. The presence of bacteria remains unobservable until bio-nose detecting them appears. We analyse situational templates characterising the process and transitions and detecting any near miss encoded as knowledge. Our facts are expiring entities at the completion of the overall process, so we might keep derived knowledge like "Nurse$_i$ has correctly assisted Patient$_j$ at 12.35 in 15 min".

We assume two doors separating the patient room and a sink in between. The nurse should enter into the anteroom, to wash hands, to wait until the air is exchanged, to apply the mask, to put clean gloves, prior to opening the second door. The complex process is human determined, the sequence is known, all single items are expected happening correctly, and sensors detect non-compliance. Let us assume that (Door$_1$ IS opened at T_0 moment) AND (Door$_2$ IS opened at T_1 moment). Simple rule for the automated reaction might be IF ($T_1 - T_0$ IS Insufficient-To-Wash-Hands) OR ($T_1 - T_0$ IS Insufficient-To-Exchange-Air) OR

(Gloves-Not-Taken-From-The-Box) OR (Humidity-Above-Sink IS Not-Changed) OR (Sink-Sensor IS Not-Reacting) OR (Any-Further-Clauses) THEN (Advert "Are Security Measures fulfilled?").

We suggest the Fuzzy conceptualisation to detail by implementation because the concrete setup is organisation dependent. The 30 s. crisp "sufficient time" becomes a Fuzzy trapezd turning a reality, especially if Chlorhexidine is used instead of water. The sensor-controlled action makes the reasonable behaviour machine-processable, while the concrete modelling remains flexible. The door opening sensors, the RFID mobility detection system and personnel identification, the humidity sensor, the reasoning server applying the rule, and the voice reaction saying something like "Sterility condition appears not satisfied?" are components of the proposed solution, where the reference embodiment is the following one. The first RFID reader is positioned to control the external door ($Door_1$) and sending data about the event, e.g. the fact about the transit and the timestamp. The second sensor is employed for controlling the hands washing, so sending the data to the server. The third sensor positioned at the internal door is controlling the second door ($Door_2$) in the same way as the first door. The knowledge server complements the central Hospital Information System elaborating the data received from sensors. It looks for abnormality patterns in behaviour calculating and applying known heuristics encoded as rules. The above-mentioned system detects a non-compliance and signals it using vocal interface.

7.4.2 Application

We build the application class supporting BMT nursing, notably a complex human-to-human process affected by human factor making it lacking the quality of service's predictability at run time, even if it is well-formalised *in guidelines* [7] and decomposable in atomic processes. The formalisation of the specific conditions gives a set of rules enabling the process control. Although human interactions are *not directly machine-processable* because of the lack of smart or appropriate sensors, inexpensive pervasive technologies, the complexity, and the uncertainty, we adopt indirect measures to monitor it against rule violations. We start from the fact that *nursing rules are tacit knowledge*, formalising it as sets of rules.

Although Reason, Norman, and Hollnagel have studied human error considering failures at execution time, how they appear in actions, and the functional characteristics or mechanisms of the human cognition, the *software detection of possible erroneous actions* remains context- dependent and not random, and it adds *complexity*. We research on *unpredictability* at run time and near-miss detection to prevent risks in BMT nursing. Combining sensing, identification and knowledge technologies for the real-time computer monitoring of potentially critical human activities, inheriting lessons learnt in other domains: automated train or aircraft's driving, and energy distribution networks, we deliver a tool to account human-to-human interactions happening in real life. The direct detection of the pathogens being practically impossible, authors configure the indirect computer-assisted measures. Starting

from the BMT nursing formalisation authors obtain the machine-processable rules, e.g. *plan recognition systems and rules for errors*. A tool proposed combines electronic sensors detecting interactions with drugs, gloves, masks, doors, sinks, and other things; a rule-based engine calculating situations/patterns, plus some context and situation awareness in process monitoring, delivering an event triggering.

In a real life there are physical objects – nurses and patients, – and non-physical objects – care processes with operations like "hand washing", "mask wearing", "patient room entering". Human brain easily processes situations and takes decisions about the elementary processes, composing the event's chain happening in real life. Our aim is to understand actions and we use computerised systems to assess human activities against the formalised representations: we wish to detect above-said situations, assess them, and take the decision about the correctness of the actions. For example, a near-miss situation might be an attempt to enter in the patient room with no mask or gloves. To qualify it we capture events and process the sequences of events, looking for a possible error, complementing best nursing strategies by inexpensive computational power. The correct sequence reflects the rules fulfilment, while the erroneous one, showing the abnormal pattern, is a near-miss candidate.

Nursing *errors stay unlikely in operations*, staff being professionally prepared, but always stay in the wrong compositions, challenging the application of the taxonomy of erroneous actions [37]. The computer-based system alone is far from the caregivers. The wireless technologies eliminate the distance between computers, caregivers, and patients, but introduce the need to identify the human actors interacting while nursing, to satisfy using Radio Frequency Identification [19, 38]. Smart wireless sensors gather biological and behavioural data from the human body, near-body area, and the ambience catching events happening in the real world. The rubber bracelets with imprinted RFID tags distinguish nurses, patients, and objects used in care processes. The combination of above-mentioned technologies brings into a new interconnected abstract world of injected real life "things" manipulated there by a computer brain doing machine reasoning, raising proactive actions, warning the personnel about near-miss risky conditions. Different sensors suit BMT nursing, including a curious patent claiming hand washing monitoring [39] – however it cannot ensure the absence of bacteria. The sequence of events forms the real-time dataflow. Sensors identify real-world objects, while computers interpret data, trigger events, and take decisions. The knowledge item's instance (sequence of events) is compared with the plan. Although the whole is not processable being too complex, it is reasonable to look for a rule violation: the unlucky event chain composes an error.

Authors start from the classification [37], *abstracting nursing as service*. An error becomes the late detected Service Level Agreement violation, a near miss is a violation detected before happening; however, many situations remain unobservable missing a detection tool. To detect a near miss or an error authors look for patterns showing phenotypes of errors: (a) intrusion, (b) replacement, (c) omission, (d) failure to complete, (e) repetition, (f) reversal, and additionally (g) wrong delay and (e) premature action, so we look for actions in wrong place, actions at wrong time,

actions of wrong type, and actions not included in the plan. Authors consider BMT nursing activities as a linear sequence of actions necessary and sufficient to reach the goal. Following Hollnagel, authors decompose the recurrent nursing process (P) in sub-processes (22).

$$P = A + B + C + D + E \ldots \tag{22}$$

In BMT nursing undertaken in isolated rooms we observe several recurrent patterns modelled using directional processes: a "safe patient room entry" as superposition of smaller inter-dependent and constrained elements, such as opening doors, washing hands, wearing masks and gloves. Any drug or food administration is accompanied by such "routines" undertaken several times per shift.

To create the baseline for reasoning we formalise the correct way to execute the intended operations and discriminate the erroneous actions, assigning them the automated reactions. The tacit knowledge contained in the "safe entry in the patient room" (Fig. 7.12) concerns the following elements: (A) opening the anteroom door, (B) wearing the mask, (C) washing hands, (D) opening the second door, (E) wearing gloves, (F) entering the patient room then. We know that the sequence of events might be right or wrong, so we have to formalise the correctness. We may look for the violation of the sequence: {A, B, C, D, E, F}. The above-decomposed process ignores the timing, but BMT nursing requires the appropriate one, e.g. we wash hands at the time t_1, we wear gloves at t_2, and not vice versa. Following Hollnagel, authors assume nursing actions being single actions that can be executed one at a time with success or failure. However, by contrast we assess the sequence of events claiming importance at the time of their occurrence and their durations, trying to catch any wrong composition. The proposed model becomes (23)

$$P(t) = A(t_1) + B(t_2) + C(t_3) + D(t_4) + E(t_5) + \cdots \tag{23}$$

Although D→B generates a potential near miss because of the "reversal" phenotype, the appropriate timing gives a key to resolve the context because all actions require "sufficient" time, and the last one is modelled. Almost all processes fill a linear sequence of events, a waterfall; however, some actions might run in parallel: air exchange runs with B and C. We formalise this empiric knowledge selecting cases to trigger by IF–THEN–ELSE rules. Swoboda [40] monitors the hand hygiene – not atomic and decomposable furthermore process – accuracy.

We represent basic nursing procedures as a collection of Fuzzy rules and check the current situations reported by the event chain against the correctness, reflected by the model. Assuming caregivers properly executing routine operations, the nursing accuracy fingerprint becomes the correctly collated sequence of uncertain elements: the hand washing timing is not constant and the system should be tuned case-by-case. Rules to tune are IF ($t_1 - t_0$ IS Insufficient-To-Wash-Hands), IF (Gloves-Are-Not-Picked-up), (Sink-Humidity IS UnChanged), plus some more additional options. Learning the nurse's usual behaviour and storing its values, the

captured nursing events are collated by computer using timing. Comparing the timing with the model, computer raises events upon violations. To avoid the personnel tracking (privacy), facts should be set as expiring entities at the completion of the overall process, limiting the system to the early warning only.

7.5 Implementation

The experimental prototype was built using the cheapest off-the-shelf sensors available from Phidgets.com. It sends data to the intelligent application server CRUAN, analysing chronology of events applying rules. The lower layer deals with the raw sensor data eliciting knowledge from the elementary information, while the higher second layer implements the decision-making features. The decision module operates with Fuzzy categories (Fig. 7.13) comparing the real life patterns with the formalised models stored in the knowledge repository: the incoming instances are compared with the formalised representations of the respective classes reflecting the nursing knowledge. Under the normal conditions the system remains silent, but any abnormal condition, reflected by rule violations, generates some feedback. The daemon component produces the reaction, aimed to interrupt the event chain happening in real life, using the traditional human–machine interface. However, an engineered real life application might adopt any other communication paradigm.

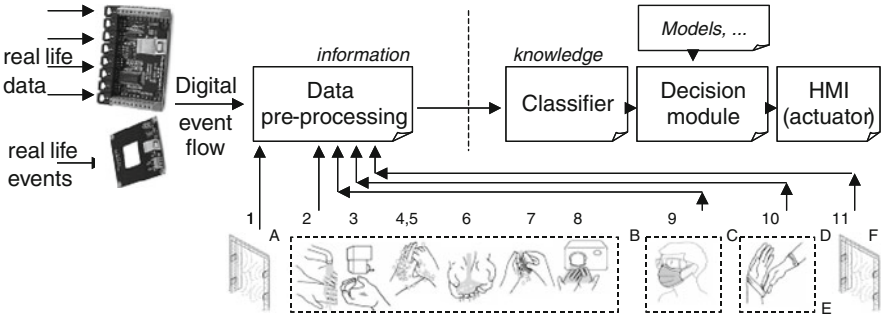


Fig. 7.13 Architectural view

7.5.1 Architecture

The architecture inherits from the research projects HPPC/SEA and IKF, and the internal mechanisms can be understood referring the application components described in [41–43]. The most important component is the use of the specific Semantic Web technologies, based on the Frame-based knowledge representation proposed and detailed in [42], while the automated uncertainty reasoning is ensured by the specialised inference engine developed by the above-mentioned IKF project.

The RDF formalisations adopted reflect the maturity level of the knowledge technologies during the IKF project; however, the framework instantiation is sufficient to solve the practical issues discussed in this work.

A possible implementation of the real life ontology-driven system for the document management and the Legacy access in natural language was described in [43]. Authors have proposed to exploit the semantic technology to efficiently communicate with the system in natural language and have illustrated a way to elicit knowledge from documents containing some domain knowledge. The work done within the Information and Knowledge Fusion (IKF) project has produced three vertical applications instantiated in different domains to validate the same architecture, in which a specific design pattern called Descriptions and Situations proposed by Gangemi and Mika [44] has been developed in order to represent procedural and regulatory knowledge. We point at the rule engine, daemons, and the invoker. Further development of the knowledge solution has been based on ontological requirement analysis, which uses competency questions [45] to be asked to experts about the use case for the application making use of the ontology. In other words, domain ontology is considered adequate to the application if it supports the tasks in the use case without over-specifying entity types. The final result is an application ontology making explicit both domain and tasks according to available software components.

The capability of the IKF platform to elicit the knowledge from semi-structured and structured sources was inherited in further developments undertaken by authors. Two software tools researched and developed within IKF – link elicitor and rule engine – become enablers of the knowledge technologies refined. The query answering engine and the event generator were based on the simple finding that since the relationship between information objects and documents is maintained fixed, the knowledge elicitation process creates the appropriate links and populates the Knowledge Repository with instances of classes describing domain-relevant objects. In this way the domain knowledge used by organisations becomes a formal collection of items indicated by relevant domain stakeholders – doctors, nurses, researchers, and so on – and processed by the knowledge systems. The tacit knowledge about the domain becomes explicit through the collection. The relevant payload becomes annotated while the relevant concepts are injected in the knowledge repository. The set of rules used for the decision-making becomes a knowledge object injected into the system, making possible the automated pattern matching between the real life situations – digitised by sensors and other devices – and the descriptions stored in knowledge base. In [46] authors illustrate a viewpoint on how to assist stakeholders of the healthcare value chain starting from the assertion about the standard doctor–patient 1-to-1 relationship. The analysis of the collaborations between healthcare professionals leads to the proposition about the proactive capability based on a “push” instead of the traditional “pull” approach, exploitable in many ways, and especially useful for advanced risk management, consequently the near-miss detection before committing the risk event.

The near-miss discovery and the timely reaction of the system permitting to interrupt the “unhappy” event chain rely on the real-time activities. The automated

system should process all the events happening in real life in the real time and should execute all the steps of the automated reasoning in real time, in order to deliver the decision and the eventual error alert in real time, before the operator will undertake the next activity. For example, the interruption of the "unsafe entry" in the patient room cannot be considered, requiring alert before beginning the above-described operation. The real-time alert delivery becomes possible if the system operates very fast and leaves some time for the feedback actuation in real time, taking a decision in "before real time", which is an important requirement limiting the use of the semantic technologies, notably requiring more time to process the events and deliver the decisions. The above-mentioned findings offer the possibility to raise events and execute some code whenever conditions described by rules are satisfied, enabling advanced considerations about the time reasoning included in the near-miss management, which is one object of this work.

7.5.2 Selection of Components

The population of the specific knowledge base reflects above-mentioned findings about the care processes, imprinted in Fuzzy rules, and some local specificity. For instance the pre- and post-intravenous infusions have the durations corresponding to the physical length of the venous lines.

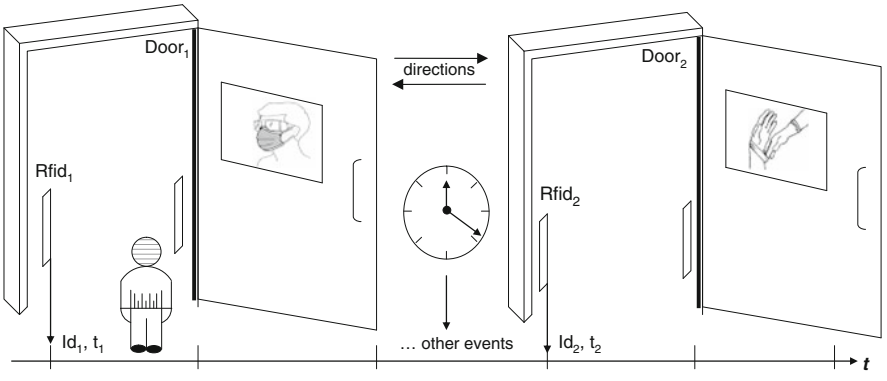


Fig. 7.14 Patient room and respective model

We equip the doors by cheap RFID readers identifying the nurses. Because the air intake occurs at one side and air exhaust occurs at the opposite side of hospital rooms (Fig. 7.14) – there are pressure differentials – the pressure sensor can be used to register the transits. Assuming the continuity of care released by different nurses along three shifts, the setup might be simplified ignoring the identification. Self-closing doors maintain constant pressure differentials among the rooms, anterooms, and hallways. Opening doors the airflow happens because of the physical phenomena, and the simple pressure sensor shows the increasing/decreasing

pressure curve, reflecting the “entering in” or “going out” events. However, the pressure sensors have bigger latency rather than RFID ones, leaving less time for the decision-making: we opt for RFID.

Further option might come from the air conditioning and ventilation (HVAC) system interfaced with the application: infection control staff works with maintenance personnel to protect BMT centres from mould spores bursts that might occur when air-handling systems are restarted after maintenance. However, the integration adds costs.

BMT centres are cleaned at least once a day with special attention to dust control. Water leaks are cleaned up as soon as possible to prevent mould proliferation. Humidity control sensors are capable to detect such conditions because of humidity curves, trespassing thresholds. System correlates RFID data identifying the cleaners – nurse assistants – with above-said humidity curves forwarding findings as events.

Staffs follow guidelines for hospital isolation and preventing nosocomial infections. Standard precautions anticipate conditions of possible contact with body fluids: hand washing, wearing appropriate gloves, surgical masks, eye/face protection, etc. We propose the use of sensors, installation of which reflects local organisation, controlling all above-said conditions, e.g. tracking workflows is indicated.

Researchers propose immune-suppressed and those undergoing conditioning therapy wearing surgical mask and gloves when exiting private rooms, minimizing the time spent in crowded areas to reduce exposure to infections. We do not track actually patients. However, a simple RFID-based wearable personal setup accounting outside stay and polluted context might identify, track, and estimate risks, without reducing them.

Hand washing is the most critical but effective infection preventing procedure. All operators should wash their hands before and after any direct contact with patients, and especially before entering and after leaving rooms where BMT and conditioning therapy patients stay [4, 8]. The proposed architecture efficiently triggers – because of the rules – any near-miss condition, so it is suggested for such complex environment. Anti-microbial soaps and liquids ensure hand hygiene [47, 48]. We account the timing of this sub-process; however, two patterns showing the water and Chlorhexidine use are totally different, adding significant unpredictability and some complexity. We have tried to compare in software the individual preferences along different shifts, adding dedicated routines, but this approach leads to the machine-learning algorithms use.

Worker puts gloves on after hand washing and discards them in the same room before washing hands again after exiting the room. Gloves should always be changed between patients and before touching a clean area. We assume this event to commit pending transactions, because odd and even events link the sub-processes. Moreover, to keep the privacy, we clean all completed loops, even if the feedback is released, assuming that the error reporting should be done using traditional instruments.

A long list of rules governing processes, where things/objects are identifiable by RFID tags, might be supported by automation. All personnel should sterilise, disinfect, and maintain equipment and devices using only registered compounds as ruled by guidelines [49–52]. Such processes are controllable compounds being a part of traceable supply chain, but the current setup does not implement it.

BMT workers have a policy [53] and recommendations regarding their immunisations and vaccinations, but automated tracking lying in Hospital Information Systems is complex because of requiring interoperability with many external entities. No support is available currently to implement this feature, and we recommend a form-based crosscheck procedure.

Bloodstream infections associated with intravenous catheters are prevented by covering them from tap water contamination, replacing devices in accordance with recommendations and performing intravenous infusions whenever possible [54, 55]. The venous line topology gives the mobility to the patient inside the room, but the length is environment specific and does require the tuning of the nursing procedures. All related parameters are hospital-specific and should not be generalised. Needleless intravenous tools impose appropriate staff training [54]. Additional training is needed because of possible interactions depending on the length of the venous line, where the local specificity reflects the tacit knowledge: aciclovir, not compatible with solutions containing proteins, requires the pre- and post-infusion washing of the venous line.

BMT patients should stay under contact precautions during nosocomial outbreaks [56]. All persons entering the BMT centre should be screened daily for symptoms. Staff members having respiratory infection symptoms are restricted and reassigned to non-patient care duties to minimise the risk [56]. Visitors with symptoms should defer their visits [56]: no efficient ICT can be set up.

ICT-assisted risk management in nursing monitors an explicit correspondence between the prescription and administration of drugs employing RFID bracelets and handheld devices simply comparing two lists and raising attention about differences. Different is the situation in BMT where the control on the fulfilment of the rules and guidelines about nosocomial infection prevention becomes essential. BMT nursing benefits from the introduction of some sensorial devices able to detect the conditions characterising risks, the knowledge engine running encoded/learnt heuristics and rules in order to detect and trigger any risk. This allows raising proactive actions adverting the personnel falling into near-miss condition. The engineering solution combines electric/electronic tools (sensors, actuators); rule-based knowledge technology-enhanced solution comprising knowledge representation, rules, intelligent heuristics, context and situation awareness, and machine reasoning; ICT with process monitoring and event triggering; and advanced human machine interfaces.

An interesting setup is mistakes avoidance help: a smart tool capable to crosscheck procedures and processes we do reducing errors in the areas where mistakes are expensive or critical. The system recognises the tagged objects, the interactions, analyses the correctness of the formalised representation, and releases warnings. In our case the doors equipped with readers and nurses wearing tagged objects release the information about the roles and give the context. In healthcare domain the drug

administration [38] and the blood compatibility check [57] tools are described. Hand hygiene monitoring idea is not unique: curious devices promising cleaned hands [39, 58] exist, even if lacking a method to detect the hygiene. The study [40] determines the effects of the hand hygiene monitoring, which is an important part of hygiene. However, preventing healthy patients from acquiring an infection holistically would be preferable, leading to the overall process compliance monitoring against the guidelines for caregivers entering BMT rooms.

7.5.3 Intelligent Layer

The most recent publication suggesting concrete implementation is the European Nursing care Pathways classification [59]. This nursing classification has been developed to illustrate the nursing care process within the context of the nursing documentation in standardised language. Its structure supports the decision-making within the context of the nursing care process through the illustration of up-to-date nursing knowledge, offering 516 formalised nursing diagnoses. Taking a class from the group of nursing objectives as an example, we have some attributes permitting to classify as "normal" or "abnormal" the real life situations characterised by data acquired by sensors. The ontology cannot help us in the selection of the appropriate sensing instruments, even if it might suggest an appropriate class to look for.

For instance the class "Breathing" from the domain of nursing diagnoses gives a category of "Physiologic inspiration and expiration", characterised by following measurable attributes (sub-categories): Breathing frequency of an adult is between 12 and 20 breaths/min, breathing frequency of a child is between 15 and 30 breaths/min, Breathing frequency of a newborn is between 30 and 50 breaths/min. Tidal volume at rest is approx. 7 ml × body weight. We can assign the additional slot explaining which sensors would be optimal, e.g. thermistor-based, nano-structured optical fibre, audio- or other sensors for breathing airflow monitoring. Knowing this we might instantiate the automated machine-reasoning process relying on the logical predicate function Assess_Breathing (24) receiving parameters Patient_Age and Breathing_Frequency.

$$\text{Assess\_Breathing}\,(\text{Breathing\_Frequency},\ \text{Patient\_Age}) = \text{Normal/Abnormal} \tag{24}$$

The function Assess_Breathing might be crisp in the simplest case or Fuzzy; however, it receives one crisp and one Fuzzy value as parameter – because of the need to define the age categories "adult", "newborn", and "child". The implementation of the above-said reasoning procedure, relying on two sensors assessing the breathing frequency and tidal volume, respectively, becomes a valuable knowledge instrument to support the clinical nursing: an automated alarm is possible each time the patient's values become "out of range". The extra effort needed to use the above-mentioned categorisation is to define correctly the trigger raising an alarm, e.g. basically the set of three thresholds (Fig. 7.15). The above-said function generates

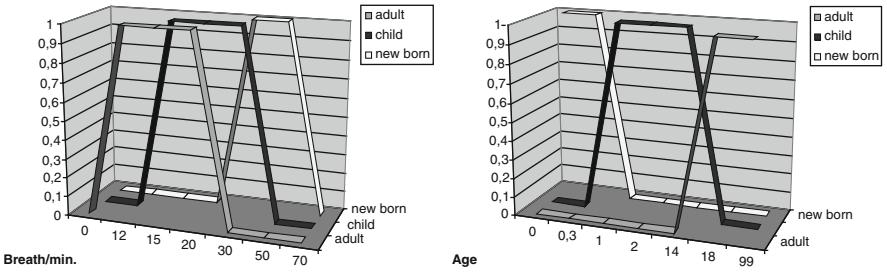


Fig. 7.15 Breathing frequency and corresponding age

an alarm condition each time the breathing frequency trespasses the fixed threshold – 25 breaths/min in adults for example – introducing also false alarms.

In a similar way we model all basic nursing processes monitored by CRUAN and represent their timing in Fuzzy terms. To avoid false alarms we add the capability to forgive some isolated out-of-range values, introducing an incremental counter and the time interval in which the observation of out-of-range values is being accumulated. The personalised assessment function ignores some isolated higher/lower values happening during – for instance 5 – last timeslots weighted as shown in (25).

$$A'(t_i) = A(t_i)+0.5\bullet A(t_{i-1})+0.33\bullet A(t_{i-2})+0.25\bullet A(t_{i-3})+0.2\bullet A(t_{i-4})+\cdots \quad (25)$$

Adopting the weights 0, 1/2, 1/3, ... assigned to the latest measurements we try to forget gradually the oldest irregular episodes. The described A' function appears efficient when nurses alternate the chemical (30) and water-based (2 min) hand washing. We propose stand-alone installation running independently from the main legacy. We clear all committed transactions after the nurse leaves the patient room at the second "opening $Door_2$" event to keep the privacy.

7.5.4 Feedback

As any risk managements system, to become a routine tool a near-miss solution should preserve the operator's privacy. The proposed computerised system is not storing warnings or error messages displaying them once on the screen: it keeps the relation computer–nurse strictly private. Should a real near miss happen, the operator might report the event using existing solutions, while the computerised support remains a trusted personal advisor. The underlying patterns stored in the knowledge base, not accessible to the end user routinely, should be deleted as well to complete the picture. Nearby the second door the LCD screen positioned shows the process representation to nurse before being entered into the patient's room. The resulting application and human–machine interface (*HMI* hereafter) paradigm adopted is shown in Fig. 7.16. The adopted graphics are sufficiently intuitive and few refinements were made after the presentation of the first version to nurses.

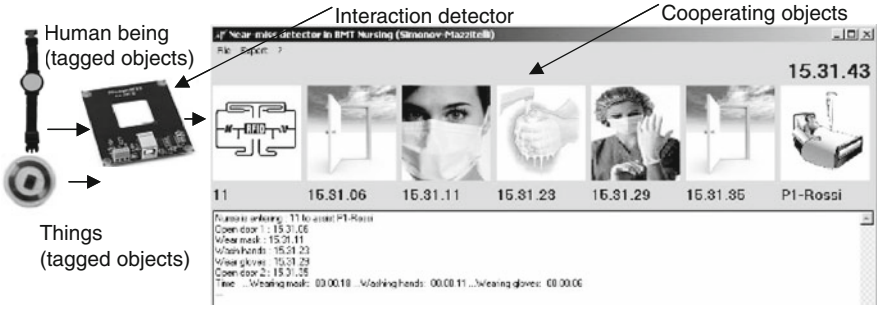


Fig. 7.16 BMT nursing application

The consensus about the usability was achieved, as well as the single error signalling. The discussion about how to signal multiple errors was more difficult. The inversion of some atomic processes might generate multiple errors to signal with more lines of warnings and messages, but nurses have argued that the single message is sufficient. The concrete on-site tuning has shown that the initial assumption about the "hand washing" timing was appropriate. However during the night and especially at the end of the night shift the traditional hand washing is frequently replaced by the specific liquids (Chlorhexidine) use, reducing the auxiliary process time, excluding the atomic sub-processes. The computerised system might show false alarms, requiring the corrective function or two modelled patterns of hand washing to compare with. An important suggestion received from nurses was to emphasise the error message, showing it "at a glance" as a big bitmap to enable to perceive visually the problem after being advised acoustically.

7.5.5 Correlations

We start from the typical day-by-day activities exemplified in Fig. 7.17. The real pattern depends on the individual care plan, the drugs used, and many other parameters, determining the event chain to be originated. On the time arrow we show events belonging to different human actors: the nurse (N) and the nurse assistant (NA). We observe that each care action is accompanied by the technical sequence of the accompanying operations described in the previous chapters.

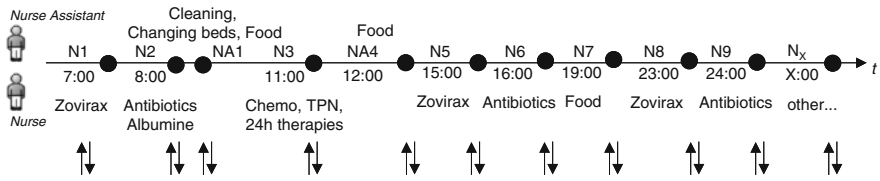


Fig. 7.17 BMT nursing care daily actions

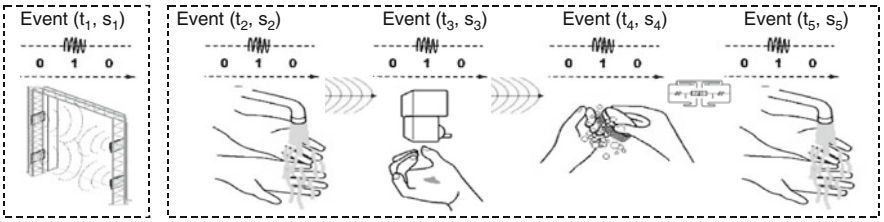


Fig. 7.18 Auxiliary processes

In BMT nursing case, a simple hand washing control system might be considered (Fig. 7.18) because a decomposable process originates events, potentially leading to the above-described phenotypes of errors. It could use a set of proximity and/or RFID sensors whose data are used to compute the "normality" or "error" patterns and then to assess the situation applying the model. Generally, Fuzzy Logic is so forgiving that the system will probably work the first time without any exceptions.

To control the "hand washing" process we define the concept dirtiness (Fig. 7.19) helping to treat different process durations, jumping from 30 s to 2 min for chemical and traditional hand washing. The association between the dirtiness and the washing time appears intuitive and modern Fuzzy-controlled wash machines use similar approach for performance optimisation.

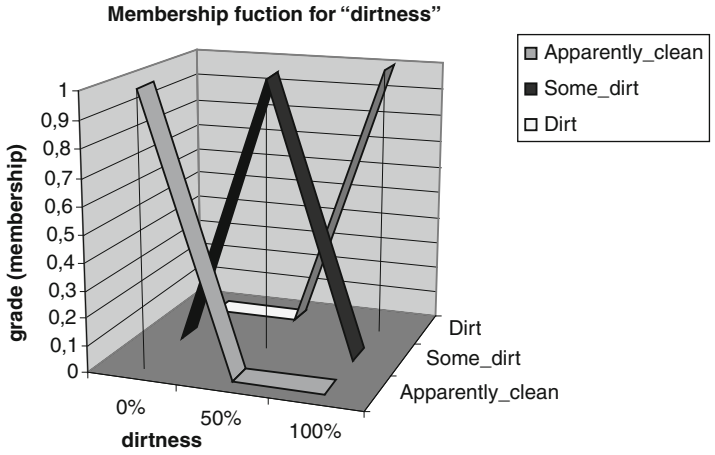


Fig. 7.19 Fuzzy definition of "dirtiness"

In BMT case the hand washing process might be measured and assessed as proposed in Fig. 7.20 using the Fuzzy definitions of the "sufficient" time. In effect, the use of proposed linguistic variables and Fuzzy rules exploits the tolerance for imprecision and uncertainty; it mimics the ability of the human mind to summarise data and focus on decision-relevant information.

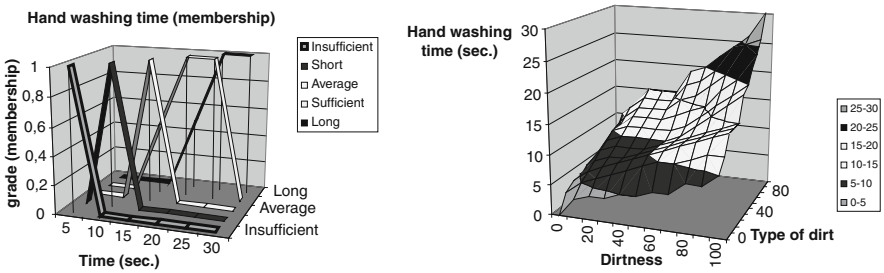


Fig. 7.20 Hand washing (timing) assessment

The adopted approach lets describe the human experience and the tacit domain knowledge in simple rules. It does not require any system modelling or complex math equations governing the relationship between inputs and outputs. In our case it significantly simplifies design complexity, because it takes only a few rules to describe the setup, which may require several lines of conventional software. In effect any care operation performed in the BMT patient room does require the full sequence of the auxiliary operations, including the hand washing one (Fig. 7.21).

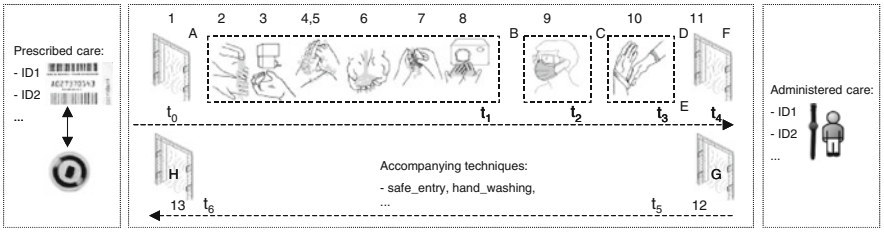


Fig. 7.21 Auxiliary processes accompanying care actions

The above findings suggest the introduction of the semantically enriched process support ICT solution tutoring the everyday nursing, which capitalises on the availability of ambient intelligence sensors, RFID, workflow control, and future bio- and nano-sensors. In CRUAN the machine reasoning operates under uncertainty [60], while the process impacted by human-to-human interaction, notably human factor, is formalised using Fuzzy rules. The decision-making process is supported by domain/process ontologies.

The correlation between the weariness and the latency in human actions is known. The latency measured in recurrent nursing actions might increase along the shift, opening an interesting opportunity because process dynamics reflects sleepiness/weariness [61, 62] for a concrete nurse. The event latency, e.g. the difference between the timing values of the same routine operation performed by the same nurse along the shift's timeline (at beginning, in-between, and at the end), compared with those referring the usual, average, or previous shift's values, enables to detect

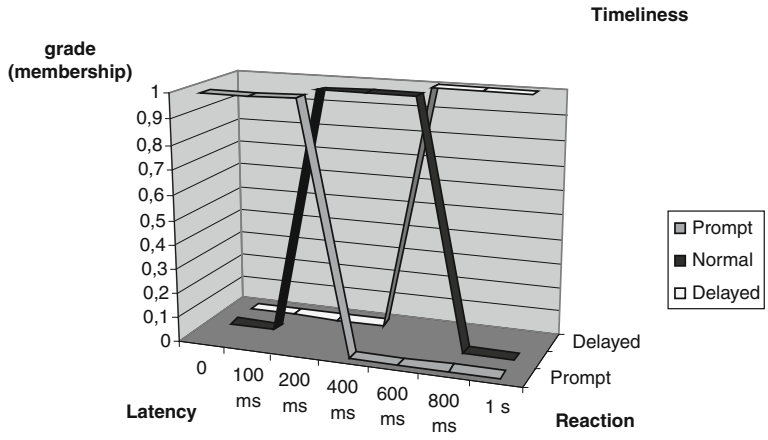


Fig. 7.22 Latency for weariness assessment

the altered latency patterns along long-lasting shifts. We characterise the nurse reaction distinguishing between the prompt, normal, and delayed ones, starting from the functions presented in Fig. 7.22.

Authors observe that the modelled timing appears strictly personal, consequently it should not be generalised. The average values calculated using the gathered statistics appears less important and would increase the false alarms.

The latency practically measured in recurrent nursing actions confirms the increasing trend manifesting along the shift, opening an interesting opportunity because process dynamics reflects the diminished vital resources caused by sleepiness/weariness [61, 62]. The event latency measured by CRUAN is the difference between the timing values of the *same* routine operation performed by the *same* nurse along the shift (at the beginning, in-between, and at the end) compared with those referring the usual/average values and enables to detect the altered latency patterns along long-lasting shifts (26).

$$\text{Latency}\ (\text{Act}_i, t_i) = \text{Dur}\ (\text{Act}_i, t_i) - \text{Avg}\ (\text{Actor}_i, [t_i', t_i'', t_{i-1}, t_{i-2}, \ldots]) \qquad (26)$$

Authors innovate adopting the timing, collating sequences, analysing the process composition rather than components, and data-warehousing latencies. The segmentation of the shift in three sub-categories is an arbitrary choice and might be replaced by other segmentation schemes. The overall picture of the latency is shown in Fig. 7.23, observing that the individual latency might be wrong, but the co-occurrence of more indicators is significant: the delayed reaction in all operations performed at 5 AM compared with the values registered at 11 AM appears indicating some weariness.

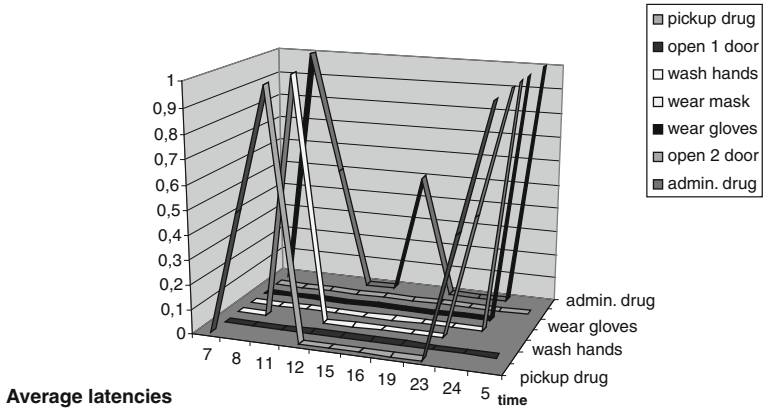


Fig. 7.23 Average latencies (in Fuzzy terms)

7.6 Conclusions

Authors have proposed an approach modelling BMT nursing through auxiliary processes accompanying main tasks and a possible way to translate nursing guidelines in rules. This work shows a possible BMT nursing analysis technique benefiting from the use of semantic information. The main idea to annotate the relevant process elements and their timing by ontological concepts enables the semantic mining and generates the main benefit: the concept-based analysis about the care situations, nurse–patient interactions captured by sensors. The proposed method depends on the local organisation and the specific timing. For instance the 4 m long venous line, ensuring the unrestricted mobility in the patient room require the different specific values for the timing of the pre- and post-infusion processes, compared with a department using a shorterone, for example, the 2 m long line. Consequently the resulting modelling may change, requiring additional personalisation being transferred to another reality. The values tuned or learned in a given BMT department remain valid only for this given topology, contributing in the system costs. The Fuzzy formalisation reflects the uncertainty explaining in a meaningful form the statements postulated by guidelines. The prototypal software solution shows the practical possibility to instantiate an application class managing near-miss conditions and the quality of nursing service.

The practical results are in line with the expectations: the system appears useful and transforms unobserved sequences into warnings, released prior to entering into the patient room. The described system embedding a limited number of rules appears capable to evaluate real life situations detecting near-miss conditions in real time. The erroneous actions are detectable using the common sensors – RFID, proximity, air pressure, humidity, etc. – and Fuzzy controllers. The adoption of modified Hollnagel phenotypes and related patterns allows satisfactory near-miss detection.

Fuzzy rules are the essential part of the solution because of the nature of human-to-human interactions. The Internet of Things technology, starting on RFID devices, becomes extremely useful. The individual behaviour gives a needed event flow to work with because of the latency dynamics (reflecting weariness) rather than value inferred by similarity clustering algorithms. A collateral result that emerged during the trial was the possibility to adopt the solution to assess empirically the weariness of nurses at the end of the night shift. The technical operations characterised by increased latencies or the case showing the reversal phenotype for an event "gloves wearing" at 5 AM demonstrate the reduced attention level, caused presumptively by the weariness. The very short "wash hands" at 2 AM appears ambiguous because some nurses do prefer using Chlorhexidine to avoid to disturb patients by running water, while an event that occurred at 6.50 AM might become a near miss with higher probability. It is reasonable to adopt the individual latency along the shift because of reflecting the operator's weariness, impacting on a human error probability.

We do not report numerical values measured in the hospital because of the above-mentioned specificities, privacy constraints, keeping the generalised correlations only. We clear all the transactions to keep the "not guilty" status protecting the privacy of operators and because of possible legal implications. Moreover it is important to note that the reliability of sensorial systems, including RFID ones, is not total, and some transactions might remain pending.

Acknowledgments Some of the achievements described in the chapter are inherited from Eureka co-funded projects HPPC/SEA, IKF, and FP7 project DOC@HAND. The work described in this chapter draws upon the contributions of many people, to whom the authors are indebted. Of course authors are solely responsible for any possible mistakes.

References

1. McCormick, K.: A concept analysis of uncertainty in illness. Journal of Nursing Scholarship 34(2) (2002) 127–131
2. Hilton, B.A.: Perceptions of uncertainty: Its relevance to life-threatening and chronic illness. Critical Care Nurse 12(2) (1992) 70–73
3. Norville, R.: Hematopoietic stem cell transplantation. In: Tomlinson, D., Kline N. (eds.) Pediatric Oncology Nursing. Advanced Clinical Handbook, Springer (2005)
4. Garner, J.: Guideline for isolation precautions in hospitals. Hospital infection control practices advisory committee. Infect Control Hosp Epidemiol 17(1) (1996) 53–80; published erratum, Infect Control Hosp Epidemiol 17(4) (1996) 214
5. Combination Chemotherapy and Bone Marrow Transplant in Treating Patients With Aplastic Anemia or Hematologic Cancer, National Cancer Institute (NCI) study, Sep. 2007
6. McCann, S. et al.: Outbreaks of infectious diseases in stem cell transplant units: A silent cause of death, Infectious diseases working party of EBMT. BoneMarrow Transplant 33 (2004) 519–529
7. Guidelines for Preventing Opportunistic Infections Among Hematopoietic Stem Cell Transplant Recipients, Recom. of Am. Soc. of Blood and Marrow Transplantation, www.cdc.gov/mmwr/preview/mmwrhtml/rr4910a1.htm. Accessed 18 Jan 2008
8. Garner, J., Favero, M.: Guidelines for hand washing and hospital environmental control. Springfield, VA, NTIS. United States Department of Commerce (1985)

9. Reason, J.: Actions not as planned: The price of automatization. In: Underwood, G., Stevens, R. (eds.) Aspects of Consciousness, Vol. 1. Academic Press, London (1979)
10. Norman, D.A.: Categorization of action slips. Psychological Review 88 (1981) 1–15
11. Norman, D.A.: Things That Make Us Smart: Defending Human Attributes in the Age of the Machine. Addison-Wesley, Reading, MA, USA. ISBN 0-201-62695-0 (1993)
12. Hollnagel, E.: The phenotype of erroneous actions: Implications for HCI design. In: Weir, G.R.S., Alty, J.L. (eds.) Human-Computer Interaction and Complex Systems. Academic Press, London, ISBN 0-12-742660-4, Chapter 4 (1991)
13. Rasmussen, J., Pedersen, O.M., Mancini, G., Carnino, A., Griffon, M., Gagnolet, P.: Classification system for reporting events involving human malfunctions. Technical Report Riso-M-2240, SIN-DOC(81)14, Riso National Laboratory, Roskilde, Denmark (1981)
14. Reason, J.: Generic error-modelling system (GEMS): A cognitive framework for locating human error forms. In: Rasmussen, J., Duncan, K., Leplat, J. (eds.) New Technology and Human Error. Wiley, London (1987)
15. Grant, T.J. 2001. Towards a Taxonomy of Erroneous Planning. Proceedings, 20th workshop of UK Planning & Scheduling Special Interest Group, University of Edinburgh, Scotland, 13–14 (December 2001), 1368–5708.
16. Eliade, M.: The Myth of the Eternal Return: Cosmos and History. Princeton University Press, Princeton, NJ (2005)
17. Reichenbach, H.: Philosophy of Space and Time. N.Y., Chapter 2.22 (1958)
18. Windelband, W.: Preludii. Filosofskie rechi i statji, SPB (1904)
19. Finkenzeller, K.: RFID Handbook 2nd Edition, Wiley, New York (2003)
20. Haikin S.: Cognitive radio: Brain-empowered wireless communications, selected areas in communications. IEEE Journal 23(2) (Feb. 2005) 201–220
21. Ivanov, B., Ruser, H., Kellner, M.: Presence detection and person identification in smart homes. In: International Conference of Sensors and Systems 80–85. St. Petersburg Technical University, Russia (2002)
22. Raducanu, B., Subramanian, S., Markopoulos. P.: Human Presence detection by smart devices. In: Proceedings of Engineering of Intelligent Systems (EIS2004), Funchal, Island of Madeira, Portugal, 29 Feb-2 (March 2004)
23. http://en.wikipedia.org/wiki/Ultra_wide_band. Accessed 18 Jan 2008
24. http://en.wikipedia.org/wiki/Terahertz_radiation. Accessed 18 Jan 2008
25. Bobrov, A.: Polevye informazionnye vzaimodejstvija, Orel (2003)
26. Zadeh, L.: Fuzzy Sets, Information and Control (1965)
27. Baldwin, J.: Fuzzy logic and fuzzy reasoning. In: Mamdani, E., Gaines, B. (eds.) Fuzzy Reasoning and Its Applications. Academic Press, London (1981)
28. Kruse, R. et al.: Foundations of Fuzzy Systems, Wiley, Chichester (1994)
29. Halpern, J.: Reasoning about Uncertainty, MIT Press, Cambridge, MA (2003)
30. Endres-Niggemeyer, B.: Empirical Methods for Ontology Engineering in Bone Marrow Transplantation, in [ISKO Germany 6 Hamburg 1999] (2000) 335–341
31. Golbreich, C., Mercier, S.: Construction of the Dialysis and Transplantation Ontology, Advantages, Limits, and Questions About Protégé OWL
32. Henry, S, Mead, C.: Nursing classifications: Necessary but not sufficient for representing "what nurses do" for inclusion in computer-based patient records. Journal of the American Medical Informatics Association 4 (1997) 222–232
33. Hardiker N, Rector A.: Modeling nursing terminology using the GRAIL representation language. Journal of the American Medical Informatics Association 5 (1998) 120–128
34. Hardiker N.: Mediating between nursing intervention terminology systems. In: Bakken S. (ed.) A Medical Informatics Odyssey: Visions of the Future and Lessons from the Past. Hanley & Belfus, Philadelphia (2001) 239–243
35. Hardiker N.: Logical ontology for mediating between nursing intervention terminology systems. Methods of Information in Medicine 42 (2003) 265–270

36. Paik, W., Ham, E.: Learning to generate an ontology-based nursing care plan by virtual collaboration. Lecture Notes in Computer Science 3762 (2005) 1200–1204
37. Hollnagel, E.: The phenotype of erroneous actions. International Journal of Man-Machine Studies 39 (1993) 1–32
38. Houliston, B.: Integrating RFID Technology into a Drug Administration System, Health Care & Informatics Review Online (Dec. 2005)
39. Taneff, Y., Rice, J.: Hand cleansing device with monitoring capability. US Patent WO/2005/055793, 23-06-2005
40. Swoboda, S., et al.: Electronic monitoring & voice prompts improve hand hygiene and decrease nosocomial infections in intermediate care unit. Critical Care Medicine 2(32) (Feb 2004) 358–363
41. Sammartino L., Simonov M., Soroldoni M., Tettamanzi A.: GAMUT – A system for customer modeling based on evolutionary algorithms. In: Whitley, L. D., et al. (eds.) Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '00), Las Vegas, Nevada, USA, July 8–12, Morgan Kaufmann, San Francisco, CA (2000)
42. Tettamanzi A.: A fuzzy frame-based knowledge representation formalism, In: LNCS: Lecture notes in Artificial Intelligence, Vol. 2955. Springer, Berlin (2006) 55–62
43. Gangemi A., Simonov M., Soroldoni M.: Ontology-driven natural language access to legacy and web services in the insurance domain. In: Abramowicz, W. (ed.) BIS 2004, Poznan, Wydawnictwo Akademii Ekonomicznej w Poznaniu (2005)
44. Gangemi A., Mika P.: Understanding the semantic web through descriptions and situations. In: Meersman, R., et al. (eds.) Proceedings of ODBASE03. Springer, Berlin (2003)
45. Gruninger M., Fox M.S.: "The Logic of Enterprise Modelling", in Modelling and Methodologies for Enterprise Integration (1996) 140–157
46. Simonov M., et al.: Information, knowledge and interoperability for healthcare domain. In: Proceedings of AXMEDIS 2005, IEEE Computer Society Washington, USA (2005)
47. Rotter, M.: Chapter 87: Hand washing and hand disinfection. In: Mayhall CG. (ed.) Hospital Epidemiology and Infection Control. 2nd ed. Lippincott Williams and Wilkins, Baltimore, MD (1999) 1339–1355
48. Larson, E.: APIC guideline for handwashing & hand antisepsis in health care settings. Am J Infect Control 23(4) (1995) 251–269
49. CDC. Guideline for prevention of nosocomial pneumonia. Respiratory Care39(12) (1994) 1191–1236
50. American Academy of Architecture for Health, Guidelines for Design and Construction of Hospital and Medical Facilities.American Institute of Architects Press, Washington, 1996–1967:58
51. Rutala, W.: APIC guideline for selection and use of disinfectants. American Journal of Infection Control 24(4) (1996) 313–342
52. National Antimicrobial Information Network.: List of EPA registered products. At www.epa.gov/oppad001. Accessed 18 Feb 2008
53. CDC.: Immunization of health care workers: Recommendations of the advisory committee on immunization practices (ACIP) and the hospital infection control practices advisory committee. MMWR 1997; 46 (No. RR-18) 1–42
54. Toscano, C., et al.: Bloodstream infections associated with needleless device use, bathing practices and home infusion. In: Proceedings of the 48th Annual Epidemic Intelligence Service Conference, April 19–23, Atlanta, GA (1999) 47
55. Do, A.N., et al.: Bloodstream infection associated with needleless device use and the importance of infection-control practices in the home health care setting. Journal of Infectious Diseases 179(2) (1999) 442–448
56. Raad, I., et al.: Infection control of nosocomial respiratory viral disease in the immunocompromised host. American Journal of Medicine 102(3A) (1997) 48–52
57. Patient safety applications of bar code and RFID technologies, White Paper, Zebra Technologies (2005)

58. Rice et al.: Hand cleaning device with monitoring capability. U.S. Patent 2005/0134465 A1, June 23 (2005)
59. Berger, S., Helmbold, A., Mosebach, H., Opel, B., Wieteck, P.: Wissenschaftliche Hintergründe ENP®. Baar-Ebenhausen (2008)
60. Bloch, I., Petrosino, A., Tettamanzi, A.: Fuzzy logic and applications, WILF 2005, vol. 3849 of Lecture Notes in Computer Science. Springer, Berlin (2006)
61. Scott, L., Rogers, A., Hwang, W.: Effects of critical care nurses' work hours on vigilance and patients' safety. American Journal of Critical Care 15 (2006) 30–37
62. Blackwell, T.: Sleepiness and Fatigue in the Medical Profession, www.utmb.edu/gme/PDF/Sleepiness_and_Fatigue_Presentation.pdf. Accessed 18 Feb 2008

Chapter 8
Toward Autonomous Mining of the Sensor Web

Peisheng Zhao, Liping Di, and Genong Yu

Abstract The Sensor Web provides an interoperable way to discover, task, and retrieve sensor data over the network. The approach to knowledge discovery from the Sensor Web needs to explore data ingestion, domain models, and data processing. This chapter leverages recent advances in autonomous control and the Semantic Web technologies to present a systematic approach in which the Sensor Web and Earth science models are annotated with semantic metadata and chained together in an autonomous way as a service chain to simulate the process of Sensor Web mining based on semantic inference. This approach significantly advances sensor data exploration with semantics in an interoperable way to improve the accuracy and timeliness of monitoring and predicting rapidly changing Earth phenomena.

8.1 Introduction

The Sensor Web generally refers to sensor networks that are Web accessible using standard protocols and interfaces. It is extremely valuable for innovative scientific research and decision-making processes to extract useful information and knowledge from the Sensor Web. A vision of the NASA Sensor Web for Earth science is to enable "on-demand sensing of a broad array of environmental and ecological phenomena across a wide range of spatial and temporal scales, from a heterogeneous suite of sensors both in situ and in orbit" [1]. To achieve this goal requires a systematic approach to exploring the domain model, data ingestion, and data processing. Effective coordination of Sensor Web mining usually is a complex process. This process consists of not only scientific hypotheses and inferences, such as identifying temporal or spatial trends and patterns, but also data acquisition and integration,

P. Zhao (✉)
Center for Spatial Information Science and Systems (CSISS), George Mason University, Fairfax, VA 22030, USA
e-mail: pzhao@gmu.edu

H. Chen et al. (eds.), *Semantic e-Science*, Annals of Information Systems 11,
DOI 10.1007/978-1-4419-5908-9_8,

such as planning and scheduling a sensor task. Such a mining process can be viewed as a complex controllable physical system. Currently, too little attention is placed on the integration of distributed sensor data management and analysis. Scientists spend considerable time finding the most appropriate data and analysis methods, dealing with the systematic and semantic heterogeneities of data and domain models, and mastering different software to derive the results. With the increasing maturation of autonomous control and the Semantic Web technologies, it is possible to automate the process of Sensor Web mining for exploiting the full potential of sensor data.

This chapter presents a systematic approach to autonomous mining of the Sensor Web within the context of Service-Oriented Architecture (SOA). In this approach, all resources are treated as Web services, and ontologies are used to annotate sensor data, interfaces of the Sensor Web, and Earth science models with semantics to increase interoperability and to further associate them with scientific problems to increase the matching capability for discovery of the best solution. With semantic matching and inference, a service chain which represents the process of Sensor Web mining can be built automatically by identifying those service instances with the most appropriate-type restrictions and running time factors. The proposed approach will significantly advance (1) the use of SOA and ontology in Earth science model integration; (2) sensor data exploration with semantics in an interoperable way; and (3) autonomous sensor data applications in Earth science to improve the accuracy and timeliness of monitoring and predicting rapidly changing Earth phenomenon.

The remainder of this chapter is organized as follows. Section 8.2 describes the Sensor Web technologies and relevant ontologies. In Section 8.3, the Earth science model and its semantic issues are discussed. The architecture and technical details of the proposed approach are described in Section 8.4. And finally, Section 8.5 presents the conclusions and plans for future work.

8.2 Sensor Web with Semantics

The term "Sensor Web" originally meant a distributed, autonomous, stand-alone, sensing activity. With the rapid development of sensor networks, the definition of the Sensor Web is currently evolving to describe a type of autonomous, taskable, and reconfigurable sensor network comprising space, airborne, and in situ sensing devices that collect measures of environmental processes, such as temperature, sound, vibration, and pressure, and deliver data products by using a set of standardized Web interfaces [2, 3]. The objective of Sensor Web mining is to extract some meaningful features from the dramatically increasing amount of Sensor Web data for a particular purpose. Therefore, how to discover, retrieve, and integrate the most appropriate sensor data becomes a challenge.

To provide a promising framework for Web-connected sensors and sensor systems accessible, controllable, and interoperable, worldwide members of the Open Geospatial Consortium (OGC) have collaborated to define, test, and document a

series of open standards in the OGC Sensor Web Enablement (SWE) activities, including

- *Observations and Measurements Schema (O and M)* [4, 5] – Standard models and XML schema for encoding observations and measurements from a sensor.
- *Sensor Model Language (SensorML)* [6] – Standard models and XML schema for describing processes and processing components of sensor systems associated with sensor discovery, location of sensor observations, measurement transformation, and listing of taskable properties.
- *Transducer Markup Language (TransducerML or TML)* [7] – Conceptual model and XML schema for describing transducers and supporting real-time streaming of data to and from sensor systems.
- *Sensor Observations Service (SOS)* [8] – Standard Web service interfaces for retrieving detailed sensor information and processes and also access to sensor observations and measurement data using spatio-temporal queries that can be filtered by phenomenon.
- *Sensor Planning Service (SPS)* [9] – Standard Web service interfaces for determining the feasibility of an intended sensor planning request, for submitting, inquiring, updating, and canceling such a request, and for requesting information about further sensor data services related the requested task.
- *Sensor Alert Service (SAS)* [10] – Standard Web service interfaces for how an alert or "alarm" from a sensor is defined, detected, subscribed, and made available to interested users.
- *Web Notification Service (WNS)* [11] – Standard Web service interfaces for asynchronous message interchange or notification among multiple services and clients.

The OGC SWE standards are getting strong international industry and government supports. By defining standardized machine-to-machine interfaces, all those standards have extraordinary significance in building interoperable applications of sensor network.

In addition, the OGC has created a catalogue service specification [12] for the discovery and retrieval of geospatial data and service metadata. In SOA, a catalogue service plays a “directory” role by helping service providers advertise the availability of resources by using meta-information and service consumers to discover the right resources by querying meta-information. Aligning the SWE standards with the OGC catalogue service provides a consistent method for the publication and discovery of sensors, sensor measurements, and sensor services. Such a catalogue service is similar to that of a mediator that provides a standard-based global schema for simplifying users’ searches by wrapping the different schemas of sensor data into catalogue information model. However, at present, sensor data is managed simply as pure data isolated from its concept domain. Data discovery is based on the exact match of keywords. Due to the lack of semantics for metadata, if a user is unfamiliar with a kind of data, data very useful for the user’s purpose might be

undiscoverable. This problem could be solved, or at least mitigated, by associating the data product's metadata with a useful meaning, i.e., including a semantic model for the Sensor Web metadata. Such a semantic model, which can be described and inferred based on ontology, would help the mining infrastructure discover the right data. Therefore, it would be desirable to describe the semantics of sensor data using ontology in metadata in order to increase interoperability as well as provide contextual information [13].

Ontology has originated from philosophy as a reference to the nature and the organization of reality. An ontology is a "specification of a conceptualization" which provides a commonly agreed understanding of domain knowledge in a generic way for sharing across applications and groups [14, 15]. To provide formal semantic descriptions of Earth science data, several projects are underway to develop a semantic framework. Described in Web Ontology Language (OWL), the ontologies within the Semantic Web for Earth and Environmental Terminology (SWEET) [16] contain several thousand terms spanning a broad extent of Earth system science and related concepts, such as those NASA's Global Change Master Directory (GCMD), Earth System Modeling Framework (ESMF), grid computing, and those of the OGC. SWEET provides a high-level semantic description of Earth system science. The ontology for geographic information metadata defined by the International Organization for Standardization ISO 19115 [17] adds the semantic meanings to the standard metadata by which data sets are explicitly associated with providers, instruments, sensors and disciplines, and the relationships among these concepts. However, the mapping between the metadata ontologies is still under investigation. A sensor ontology is built to define the concepts and properties adapted in part from SensorML, Institute of Electrical and Electronics Engineers (IEEE) Suggested Upper Merged Ontology (SUMO), and ISO 19115 as a sensor knowledge repository for synergistic fusion of heterogeneous sensor data [18]. The project "Sensor Standards Harmonization" supported by National Institute of Standards and Technology (NIST) develops a common sensor ontology based on IEEE 1451, ANSI N42.42, the Chemical, Biological, Radiological, and Nuclear (CBRN) Data Model, and the OGC SWE languages. However, for the Sensor Web mining, the existing ontologies still need to be elaborated to associate the distributed heterogeneous sensor data sets with domain concepts, allowing users to easily discover the relevant data through navigating the domain space. In addition, the ontology would also provide sufficient information so that by defining the hierarchy of sensor data types, subtle differences could be detected among similar data in the actual data mining or the interpretation of the results. The issues of the domain link and sufficient information are addressed in the proposed approach.

8.3 Semantically Enabled Earth Science Model Service

In the past few years, many Earth science models have been developed to provide efficient ways to extract useful information and derive knowledge from a large amount of raw sensor data. Each model has its own unique application area and

computational and data requirements. Generally, an Earth science model can be characterized by

- *Mission*: the scientific field of the model, for example, atmospheric composition, climate variability and change, or disaster monitoring.
- *Methodology*: the exact purpose of the model, for example, classification, prediction, clustering, detection, or association.
- *Algorithm*: the type of algorithms used in the model, for example, neural or Bayesian network.
- *Data source*: the properties of model inputs, not only the physical characteristics of the data but also the scientific significance. The Sensor Web provides discoverable, accessible, and interoperable data sources for Earth science models.
- *Result*: the properties of model outputs, for example, data format, data projection, and data resolution.
- *Quality*: the quality measurements of the model, for example, running time, accuracy, and spatial or temporal extent.

Earth science research is multiscale and multidisciplinary in nature and is data, information, and computationally intensive. Multiple models may be needed to represent the complicated processes of Earth phenomena and solve relevant scientific problems. Therefore, Sensor Web mining needs not only interoperable data but also interoperable Earth science models. Since Web service technologies provide a promising mechanism for application interoperability, it is desirable to transform heterogeneous Earth science models into Web services. The OGC defines the Web Processing Service (WPS) [19] which provides a set of standard interfaces for domain-specific computational models from simple spatial calculations to global climate change. Representing Earth science models as Web services or OGC WPSs makes it possible to build service chains to enable users to do complex data analysis in a standard way over the Web regardless of the low level of system heterogeneities. In recent years, an increasing number and amount of geospatial models have been available online as Web services, significantly enhancing the possibility of collecting and analyzing geospatial data in an interoperable manner across systems. NASA's Earth Science Gateway (ESG)[1] allows users to streamline access to science and research products, including data, models, and visualizations provided by a variety of national and international organizations, through open standard Web protocols. GeoBrain Geographic Resource Analysis Support System (GRASS) Web Services,[2] based on the functionality modules from GRASS, provides geospatial data management, raster image processing, spatial modeling and analysis, graphic/map producing, and data visualization over the network. The Algorithm Development and Mining (ADaM) Web Services[3] leverage the ADaM

[1] Earth Science Gateway, http://esg.gsfc.nasa.gov

[2] GeoBrain GRASS Web Services, http://geobrain.laits.gmu.edu:81/grassweb/manuals

[3] ADaM Web Services, http://datamining.itsc.uah.edu/adam/

toolkit to enable mining remotely sensed and other scientific data dynamically over the network for pattern recognition, image processing, optimization, and association rule exploration.

Given the many Earth science models available, solid knowledge about Earth science models is required to identify the most appropriate models for a given scientific problem. However, most users lack such knowledge. In order to simplify the development of applications in Sensor Web mining, it is necessary to associate Earth science models with the domain of scientific problems, i.e., to offer a reference model of Earth science models available for solving a given scientific problem. Ontology is usually used for capturing domain knowledge in a generic way and provides a commonly agreed understanding of a domain [14]. In the data mining context, ontology is viewed as a formal structure which encapsulates the semantics of data mining techniques and data sources and relates them to the concepts within disciplines. An ontology has been developed for the data mining domain in order to simplify the development of distributed knowledge discovery [20]. This ontology offers a reference model for different kinds of data mining tasks, methodologies, and software to help users find the most appropriate mining solution. This ontology is oriented the general data mining problems so that it only conceptualizes the generic data mining techniques and does not consider domain concepts and data. For more complex scientific problems, it is necessary to relate the data to the domain concepts by using ontology [21]. Users can then easily retrieve the relevant data sets to be compared by navigating the ontology. Applying ontology to the conceptualization of Earth science models not only provides a common language for representing features of and classifying components of Earth science models but also adds explicit meanings and relations for inference and interoperability, i.e., conceptual links among sensor data, Earth science models, and scientific problems. Ontology for Earth science models enables semantic search (concept-based) and conceptual inference to satisfy the users' requirements/needs for the complex process of Sensor Web mining. Thus, users can easily find the most appropriate Earth science model for their scientific questions without needing to go into the details of individual models.

8.4 Semantic Web Service-Based Approach

Sensor Web mining is a complex process that usually involves integrating a variety of Earth science models and sensor data. It would be desirable to have facilities to solve the issues of characterization, discovery, accessibility, and interoperability of domain models and data in an open, distributed, and heterogeneous environment. Mining infrastructure has been identified as one of the main challenges at the system level for the Sensor Web. However, most current Sensor Web research projects are still focusing on isolated individual hardware and software components that are running in their own proprietary environments. The ultimate goal of mining infrastructure is to support autonomous mining of heterogeneous sensor data to facilitate the integration of Earth science models and sensor data. This "integration" requires

facilities for finding and obtaining the mining resources, dealing with multiple data formats and operation interfaces, supporting a complex workflow process, and managing the computational intensity inherent in Sensor Web mining. The emerging SOA enables cross-platform interoperability by using loosely coupled and interoperable Web services to implement system requirements. All the services within the SOA are independent, with self-described interfaces so that they can be accessed in a standard way without knowledge of how the service actually performs its tasks. Using the SOA for construction of the mining infrastructure would provide a standard interoperable way to deploy Earth science models and sensor data as Web services, publish, discover, and invoke mining resources for individual uses and orchestrate sophisticated mining processes as service chains to synthesize useful knowledge from data automatically. Using recent advances in autonomous control and Semantic Web technologies, an ontology-driven infrastructure within the context of the SOA as Fig. 8.1 is being developed to automate sensor data acquisition and mining by seamlessly integrating heterogeneous Earth science models and the Sensor Web. The infrastructure components are described in detail in the following sections.

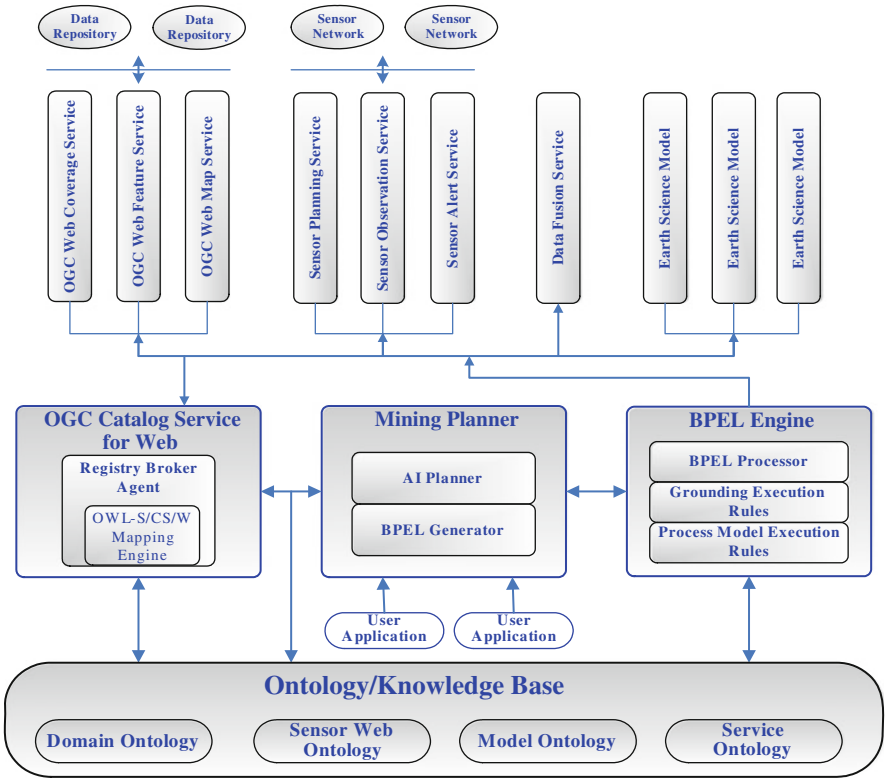


Fig. 8.1 Ontology-driven Sensor Web mining infrastructure

8.5 Process of Sensor Web Mining

Sensor Web mining in the proposed approach is viewed as an AI planning problem which begins with Earth science goals of "what," "where," and "when" and a set of decomposed sub-goals. A goal, or set of sub-goals, is further decomposed into a sequence of actions. Thus, the automated process of Sensor Web mining is designed as a set of states of a finite state machine in which the actions with an I/O behavior-based control structure change to go from one state to another. There are three atomic actions: data acquisition, data access, and data processing. All actions are organized consequently where, for each adjacent pair of processes, the execution of the first process is necessary for the execution of the second service. Figure 8.2 shows the process of Sensor Web mining.

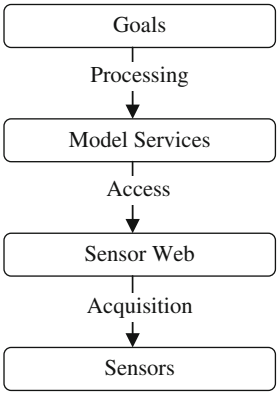


Fig. 8.2 Process of Sensor Web mining

The goals are high-level specifications of geospatial knowledge derived from the incremental development and refinement of models for monitoring, simulating, and predicting Earth processes. Typically, a goal is represented as a "what is" or "what if" question for a certain location at a certain time. For example, what the landslide susceptibility in California has been in the last 2 years and what the damage loss is if a fire occurs near an airport. A goal can be viewed as a feature vector containing, but not limited to, the following attributes [22]:

- *Mission*: what is to be measured (e.g., landslide susceptibility and damage loss).
- *Location*: where the measurements are to be taken (e.g., California and an airport).
- *Time*: when the measurements are to be taken (e.g., the last 2 years and now).

The values of these attributes may be assigned statically or obtained during process building. A goal may be decomposed into a set of sub-goals dynamically if no model service or data can satisfy it directly. For example, to calculate landslide susceptibility requires slope, aspect, and Normalized Difference Vegetation Index

(NDVI) data. If there is no such data available, the slope, aspect, and NDVI then would be sub-goals.

For a given goal, a data processing action refers to invoking the appropriate Earth science model services to extract information and derive knowledge from raw data.

A data access action is to retrieve real-time data that is distributed and archived from the Sensor Web. Typically, data access actions consist of building standard requests to invoke the Sensor Web and then stream data directly or extract the data location for data processing actions.

A data acquisition action abstracts from two notions [23] as follows:

- *Measurement*: a key-value pair consisting of a phenomenon of interest (measurement key) and an associated value.
- *Observation*: an event that triggers a scheduling sensor task to get measurements.

A data acquisition action provides the Sensor Web a collection of measurements for a location over a time period.

Furthermore, some control structures are imported to coordinate these data actions within a mining process which are as follows:

- *Condition*: associates a Boolean expression with whether an action is to be executed.
- *Choice*: consists of an ordered list of two or more possible transitions, where only one will be executed.
- *Loop*: supports repeated execution of a specified iterative action until a given condition no longer holds true.
- *Synchronization*: imposes an execution order on a set of dependent actions. For example, an action needs to wait for notification that a sensor task has been completed and the measurement is ready.

8.6 Ontology-Based Knowledge Base

The knowledge base provides the overall knowledge about Sensor Web mining. To form the conceptual structure of a knowledge base with semantic instances, a set of ontologies as Fig. 8.3 are embedded to represent the conceptualization of sensor, the Sensor Web, and Earth science domain and further used to facilitate to build the process of Sensor Web mining.

The general ontology is the core upper level vocabulary for all human consensus concepts. It defines common terms to which all other ontologies refer. The Dublin Core Metadata[4] and OpenCyc[5] are the bases of definitions of upper level concepts and assertions about these concepts.

[4]Dublin Core Metadata, http://dublincore.org

[5]OpenCyc, http://www.opencyc.org

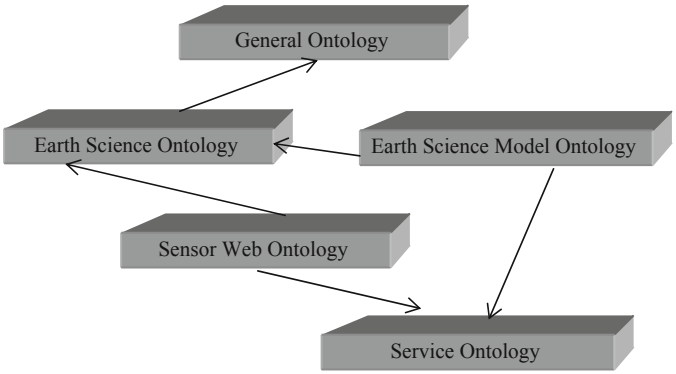


Fig. 8.3 Ontologies for Sensor Web mining

The Earth science ontology aims at providing the core concepts and knowledge structure of Earth science. It represents the problem space over which the user will search. Other ontologies directly or indirectly incorporate this ontology. By incorporating the terms in the GCMD and the ESMF, this ontology encodes the knowledge about (1) spatial–temporal factors, e.g., location, time, and units; (2) physical facts, e.g., physical phenomena, physical properties, and physical substances; and (3) disciplines, e.g., scientific domains and projects. For example, the following ontology shows "surface water" belongs to "hydrosphere," and "river" is a kind of "surface water":

```
<owl:Class ref:ID="Surface_Water">
   <rdfs:subClassOf>
       <owl:Class rdf:resource="#Hydrosphere"/>
    </rdfs:subClassOf>
</owl:Class>
<owl:Class ref:ID="River">
   <rdfs:subClassOf>
       <owl:Class rdf:resource="#Surface_Water"/>
    </rdfs:subClassOf>
</owl:Class>
<owl:Class ref:ID="Stream">
   <rdfs:subClassOf>
       <owl:Class rdf:resource="#Surface_Water"/>
    </rdfs:subClassOf>
</owl:Class>
```

The Earth science model ontology directly incorporates the Earth science domain ontology to represent the features of the available Earth science models, classify their structures, document the relationships and the constraints among them, and associate the models with scientific problems and relevant data sources. A model can

be implemented incrementally as services by using different languages on different platforms. It actually represents a collection of services – service type. The following ontology describes a wildfire spread simulation model which is designed to use fuel data, terrain data and weather data to calculate the time of spread, rate of spread and spread direction of a wildfire based on a wildfire mathematical model [24]:

```
<Service_Type rdf:ID="Wildfire_Spread_Simulation">
    <isFor><Mission rdf:ID="Wildfire"/></isFor>
    <method><Methodology rdf:ID="Simulation"/></method>
    <use> <Algorithm
    rdf:ID="Rothermel_Wildfire_Mathematical_Model"/></use>
    <input><Data_Source rdf:ID="Fuel_Data"/></input>
    <input><Data_Source rdf:ID="Terrain_Data"/></input>
    <input><Data_Source rdf:ID="Wheather_Data"/></input>
    <output><Result rdf:ID="Wildfire_Spread_Direction"/></output>
    <output><Result rdf:ID="Wildfire_Time_Of_Spread"/></output>
    <output><Result rdf:ID="Wildfire_Rate_Of_Spread"/></output>
    <quality><Quality rdf:ID="Wildfire_Model_Quality"/></quality>
</Service_Type>
```

By leveraging current sensor ontology efforts of the OGC, the World Wide Web Consortium (W3C), and the NIST, the Sensor Web ontology presented in the following example provides enhanced descriptions and meanings to the Sensor Web as well as spatial, temporal, and platform metadata with semantics for distributed heterogeneous sensor data resources:

```
<owl:ObjectProperty rdf:ID="belongTo">
    <rdfs:range rdf:resource="#Domain"/>
    <rdfs:domain rdf:resource="#Data_Type"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="usedFor">
    <rdfs:range rdf:resource="#Physical_Fact"/>
    <rdfs:domain rdf:resource="#Data_Type"/>
  </owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="extent">
    <rdfs:range rdf:resource="#Space_Temporal"/>
    <rdfs:domain rdf:resource="#Data_Type"/>
  </owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="measuredFrom">
    <rdfs:domain rdf:resource="#Data_Type"/>
    <rdfs:range rdf:resource="#Platform"/>
  </owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasInstruments">
    <rdfs:domain rdf:resource="#Platform"/>
    <rdfs:range rdf:resource="#Instrument"/>
  </owl:ObjectProperty>
```

```
<owl:ObjectProperty rdf:ID="hasSensor">
    <rdfs:range rdf:resource="#Instrument"/>
    <rdfs:domain rdf:resource="#Sensor"/>
  </owl:ObjectProperty>
```

The Sensor Web ontology directly incorporates the Earth science domain ontology to link the sensor data with scientific problem space so that users can locate data without knowing the exact metadata keywords used by the data provider, because the terms in the query may have an equivalent definition in the Earth science domain components. To provide a unified view of metadata, semantic relationships among terms in different metadata standards, such as IEEE 1451, ANSI N42.42, and OGC SWE, are also determined. Thus, there is no distinct boundary between different metadata standards. The client can use any term from any one of the metadata standards to query the data described in any one of the metadata standards.

The service ontology conforming to OWL-based Web Service Ontology (OWL-S)[6] incorporates the Earth science model ontology and the Sensor Web ontology to advertise relevant services with explicit semantics to make services machine understandable. The "profile" of OWL-S, which describes who provides the service, what the service does, and other properties of services, allows the knowledgebase to infer whether or not a particular service is appropriate to a given problem. The OWL-S "process model," which states the inputs, outputs, preconditions, and effects of a service, enables the knowledgebase to determine whether or not a service meets the requirements and conditions for invoking it. The "grounding" of OWL-S, which presents the ports, protocols, and encoding of invocation, tells the knowledgebase how to invoke a service. The semantic service description makes it possible to implement autonomous service chaining.

Since all of the ontologies are encoded in OWL and OWL-S, the inference engine in the knowledgebase is an OWL reasoner built on Prolog in which ontological information is converted into RDF triples. The engine uses built-in axioms to find all relevant entailments that may not be directly in the subclass relationships, such as the inheritance relation between classes. In addition, some domain-specific rules are defined using Semantic Web Rule Language (SWRL)[7] to derive additional knowledge from semantically annotated Earth science models and sensor data.

8.7 OGC Catalogue Service for Web (CS/W) with Semantic Augmentations

Catalogue services are very important for the discovery of data and services in each step of Sensor Web mining. The OGC CS/W is the de facto standard for supporting the discovery of and binding to registered spatial information resources. Its

[6] OWL-based Web Service Ontology, http://www.daml.org/services/owl-s/1.1

[7] SWRL: A Semantic Web Rule Language Combining OWL and RuleML, http://www.w3.org/Submission/SWRL

information model, based on the Electronic Business Registry Information Model (ebRIM) [25], defines a general model for metadata management. It is intended to provide comprehensive facilities to represent geospatial metadata and conceptual schemas. The schemas describe the kinds of objects stored in the catalogue and how these objects and the relevant descriptive information are organized and interpreted. Because of its capabilities for publishing, discovering, and accessing spatial data and services at run-time and its extensibility to supporting new object types, OGC CS/W is used in the proposed approach. In order to represent the semantics of data and services to support flexible semantic matching in the catalogue, the ontologies represented by OWL and OWL-S are embedded into ebRIM by using ebRIM's "*Classification,*" "*ClassificationSchma,*" "*ClassificationNode,*" "*Slot,*" and "*Association*" to record the corresponding OWL class, properties, and related atoms [26].

By using the ontology, the matching process can make inferences using the subsumption hierarchy to recognize semantic matches regardless of syntactic differences. Assume that a service or data is G and the request is R. The "flexible semantic matching" can deal with the following cases:

1. *Exact*, if $G = R$, then G and R are exact equivalent;
2. *Plug-in*, if G subsumes R then G could be plugged in place of R;
3. *Subsume*, if R subsumes G, then G just completes part of R and R needs other G to implement the other part of R or whole R;
4. *Failed*, there is no relationship between G and R.

Flexible semantic matching provides three types of semantically augmented search functions [27]:

- *Dataset Search*: semantically matched data type as the additional search condition in the standard OGC CS/W data set query.
- *Service Search*: semantically matched service type with optional associated data types as additional search conditions in the standard OGC CS-W service query.
- *Data-Type Driven Service Chaining*: service composition based on semantic matching, either between two services where the output of the first service provides the input of the second service or between data and services where the data provides the necessary inputs for the services.

8.8 Geospatial Web Services

8.8.1 Data Services

All data should be virtually online to allow rapid access by the mining system. A data service portal provides a common data environment that incorporates the catalogue service to make the online archived data available through the OGC Web

Coverage Service (WCS) [28], Web Feature Service (WFS) [29], and Web Map Service (WMS) [30] and real-time data available through the OGC SPS, SOS, and SAS. The OGC WCS provides an interoperable way of accessing multi-dimensional and multi-temporal sensor data as coverages to meet the requirements of client-side rendering, multi-valued coverages, and inputs to scientific models. The OGC WFS supports the networked interchange of geographical vector data as features encoded in Geography Markup Language (GML) to meet the requirements of complex spatial analysis. The OGC WMS supports the networked interchange of geospatial data as a map, which is rendered dynamically from real spatial data as a spatially referenced pictorial image, such as PNG, GIF, or JPEG, for easy display or overlay with other geospatial data. The OGC SPS enables a client to determine the feasibility of plans for a sensor and provides access to the data collected by the requested task. The OGC SOS enables a client to access heterogeneous sensor data in a standard way by leveraging the OGC O&M specification for sensor observations and the OGC TransducerML and SensorML specifications for sensors and sensor systems. The OGC SAS covers the process of alert publication, subscription, and notification to enable a client to register for and receive sensor alert messages. By providing the OGC Web services, the data service portal allows seamless access to geospatial data in a distributed environment with standard interfaces, regardless of the low level of accessing details. Thus, it becomes easy to feed the geospatial data into Earth science models.

8.8.2 Data Fusion Services

An important component of the proposed approach is the ability to fuse data or combine evidence from multiple mining activities in order to provide data that is more appropriate and a more accurate picture of the result. This can be supported by the development of a set of data fusion services:

- *Web Coordinate Transformation Service (WCTS)*: transforms coverage data from one map projection to another projection.
- *Web Clipping Service (WClS)*: allows users to get data inside/outside of a boundary expressed as the name of a location, a series of coordinate values, or a file encoded in GML polygon.
- *Web Format Translation Service (DFTS)*: takes a supported data format and converts it into any format specified by the user.

8.8.3 Earth Science Model Services

One of the most distinguishing characteristics of the proposed approach is that Earth science models are wrapped as Web services to make them describable, discoverable, chainable, and accessible in the mining system. Theoretically, one can either develop an Earth science model as a Web service from scratch or create adapters

that make the legacy system compatible with the Web services. For example, a developer may create an HTTP adapter to convert the input and output between Simple Object Access Protocol (SOAP) and Hypertext Transfer Protocol (HTTP) messages or may develop a Java adapter to call the C/C++ component through JNI (Java Native Interface). While designing the application has greater freedom and flexibility in the former case, the latter case is of greater interest since it allows one to reuse the existing code base. The issue that needs to be addressed in the latter case is whether to expose every method of the models or to combine some methods of the models into one interface to perform higher level tasks. Solving this issue depends on a given problem itself and needs to balance the efficiency and complexity of service implementation.

8.9 Mining Planner

The power of Sensor Web service, however, is not just in its standard interfaces, but also in the potential of being chained with others into a composite service to solve complex problems. It is feasible to assemble individual Earth science models and Sensor Web services into a service chain for the mining of the Sensor Web. The ontologies in the knowledgebase give the data and services explicit meanings and make the contents transferred between the services machine understandable, permitting automatic service chaining. In addition, the reasoning based on the ontologies can utilize users' requirements to suggest actions and information sources, e.g., locating data sources on a specific topic and finding Earth science models for a desired algorithm, a particular method, or a specific task.

The Mining Planner based on the AI planning algorithms is essential for autonomous Sensor Web mining. It is designed to use backward chaining methods for translating the entire problem to be solved recursively into sub-problems that can be solved by available mining services. It exploits graph structures and state transitions to extract heuristics with the constraints of geospatial contents to simulate the processes of Sensor Web mining and the construction of executable service chains. This Mining Planner supports two levels of chaining:

- *Concept chain*: a logic model in which the most appropriate service types are identified and connected logically.
- *Physical chain*: an executable service chain in which each main component is an instance of the corresponding service type in the concept chain.

For the concept chain, a graph is used to find one or more sequences of service types whose inputs and outputs are matched semantically. The choice among various paths depends on semantic matching [31]:

$$\begin{aligned} Match_{plug\text{-}in\text{-}dataType}(\mathcal{DS}, \mathcal{DQ}) = \mathcal{D}_{\mathcal{DS}} \sqsubseteq \mathcal{D}_{\mathcal{DQ}} \wedge \mathcal{PF}_{\mathcal{DS}} \sqsubseteq \mathcal{PF}_{\mathcal{DQ}} \\ \wedge\ \mathcal{P}_{\mathcal{DS}} \sqsubseteq \mathcal{P}_{\mathcal{DQ}} \wedge \mathcal{ST}_{\mathcal{DS}} \sqsubseteq \mathcal{ST}_{\mathcal{DQ}} \end{aligned} \quad (1)$$

$$Match_{plug\text{-}in\text{-}serviceType}(\mathcal{SS},\mathcal{SQ}) = \mathcal{MI}_{\mathcal{SS}} \sqsubseteq \mathcal{MI}_{\mathcal{SQ}} \wedge \mathcal{ME}_{\mathcal{SS}} \sqsubseteq \mathcal{ME}_{\mathcal{SQ}} \wedge \mathcal{A}_{\mathcal{SS}} \sqsubseteq \mathcal{A}_{\mathcal{SQ}} \wedge \mathcal{I}_{\mathcal{SQ}} \sqsubseteq \mathcal{I}_{\mathcal{SS}} \wedge \mathcal{O}_{\mathcal{SS}} \sqsubseteq \mathcal{O}_{\mathcal{SQ}} \quad (2)$$

A concept "*Data_Type*" $\mathcal{DS}$ is assumed to be a plug-in match for a requested "*Data_Type*" $\mathcal{DQ}$ (1) if all properties of the concept, "*Domain*" $\mathcal{D}_{\mathcal{DS}}$, "*Physical_Fact*" $\mathcal{PF}_{\mathcal{DS}}$, "*Spatial_Temporal*" $\mathcal{ST}_{\mathcal{DS}}$, and "*Platform*" $\mathcal{P}_{\mathcal{DS}}$ subsume their counterparts in the $\mathcal{DQ}$, i.e., $\mathcal{D}_{\mathcal{DQ}}$ $\mathcal{PF}_{\mathcal{DQ}}$ $\mathcal{ST}_{\mathcal{DQ}}$, and $\mathcal{P}_{\mathcal{DS}}$. A concept "*Service_Type*" $\mathcal{SS}$: $(\mathcal{I}_{\mathcal{SS}}, \mathcal{A}_{\mathcal{SS}}, \mathcal{MI}_{\mathcal{SS}}, \mathcal{ME}_{\mathcal{SS}}) \rightarrow \mathcal{O}_{\mathcal{SS}}$ is then assumed to be a plug-in match for a requested "*Service_Type*" $\mathcal{SQ}$: $(\mathcal{I}_{\mathcal{SQ}}\mathcal{A}_{\mathcal{SQ}}$ $\mathcal{MI}_{\mathcal{SQ}}$ $\mathcal{ME}_{\mathcal{SQ}})$ $\rightarrow$ $\mathcal{O}_{\mathcal{SQ}}$ (2) if the "*Data_Type*" of the concept input $\mathcal{I}_{\mathcal{SS}}$ is more generic, the "*Data_Type*" of the concept output $\mathcal{O}_{\mathcal{SS}}$ is more specific, and the "*Mission,*" "*Methodology,*" and "*Algorithm*" of the concept $\mathcal{M}_{\mathcal{SS}}$ are more specific than their counterparts in the $\mathcal{SQ}$. Moreover, performance is also critical to the concept chain. The length of the logical path is one criterion used to help select a plan.

To get a physical chain from a concept chain, each "*Service_Type*" in the model is mapped into a service instance. Data services are joined at the leaf nodes of a concept chain to provide the ultimate data sources. If necessary, fusion services are used to coordinate the preconditions and results of adjacent services. Each service is associated with a "*Service_Type*" and has been related to "*Condition*"s, and each data item is associated with a "*Data_Type.*" In "Condition" matching, the *Plug-in Effect-Pre* Match is of particular importance . The *Plug-in Effect-Pre Match* holds when the precondition of service *S* is more general than the effect of service *Q* (3):

$$match_{Plug\text{-}in\text{-}Effect\text{-}Pre}(\mathcal{S},\mathcal{Q}) = \mathcal{S}_{pre} \Rightarrow \mathcal{Q}_{effect} \quad (3)$$

This match means that service *S* can be connected with service *Q* without considering the issues of data representation. If no service match is found, an additional data fusion service may have to be provided in the service chain to establish this match.

8.10 BPEL Engine

The widely used Business Process Execution Language (BPEL)[8] is adopted to represent a physical service chain in the proposed approach. Compliant with mainstream standards about Web services; the BPEL Engine is a Web-based workflow engine designed to deploy, manage, and execute individual services and composite service chains written in BPEL, as in Fig. 8.4. Invocation of Web services is not limited to those that are SOAP based. The engine is capable of handling different styles of Web services. It has built-in support for invocation of Web services using different standard methods, including HTTP Get, HTTP POST, SOAP, and Java Remote

[8]Web Services Business Process Execution Language Version 2.0, http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

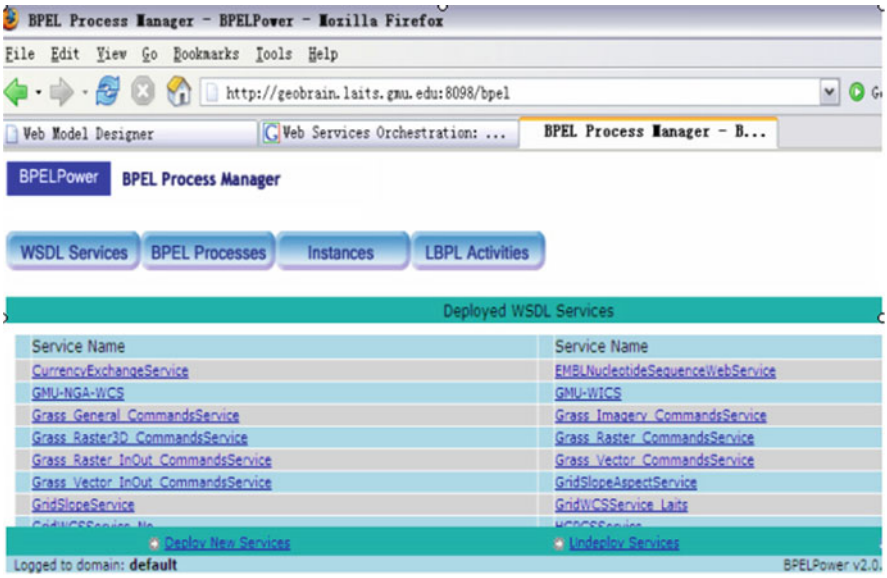


Fig. 8.4 BPEL process manager

Method Invocation (RMI). This facilitates its support of the new generation of Web services in a broad sense, for example, the newly recognized style Representational State Transfer (REST) Web service. REST utilizes the HTTP protocol, including GET, POST, DELETE, and UPDATE, and interprets the service in terms of a state machine. The deployed workflow can be thought as a standard Web service since it is published to support multiple bindings, including a SOAP binding and an HTTP binding. The HTTP-based binding compiles to REST-style Web services without the requirements SOAP messaging has. The engine has been evolved to be much more robust than many other contemporary engines when handling the complex schemas used in OGC Web services, especially GML. WS-Addressing[9] is used to support the invocation of asynchronous Web service and publish the service chain as an asynchronous Web service. Moreover, the engine supports invocation of security Web services at the transport and message levels.

8.11 Conclusions

This chapter presents a promising approach to autonomous Sensor Web mining. The most significant innovations of the proposed approach are the use of Semantic Web services to bridge the growing interoperability gap between sensor data and Earth science models and importing path planning to automatically create a concept chain to simulate the process of Sensor Web mining and a physical chain to associate the

[9]Web Services Addressing (WS-Addressing), http://www.w3.org/Submission/ws-addressing/

given problems with the real world. In this approach, domain concepts and problems are well defined by the domain ontology as the basic knowledge, sensor data, and Earth science models are well described and associated with domain space by these concepts and served by OGC Web services and Semantic Web services to make them discoverable, retrievable, and chainable in an interoperable manner. This approach provides a major mechanism for scientists and decision makers to fully and easily exploit the potential of sensor data.

Further studies will focus on investigating chain correctness and quality and on refinement of ontologies and inference rules. It is impossible to exhaustively put all relevant concepts into ontologies and represent all possible domain knowledge as rules. It is a long, evolving process. We will consequently follow the development of geospatial ontologies and standards to elaborate the mining ontologies and inference rules for sketching geospatial space precisely. Currently, the mining chain is built upon ontology matching and the length of the planning path without considering service creditability and quality. Thus, a trust mechanism, such as quality ranking, needs to be introduced in the further development.

References

1. Moe, K., Smith, S., Prescott, G., Sherwood, R.: Sensor web technologies for NASA earth science. In: Proceedings of the 2008 IEEE Aerospace Conference, IEEE Press, New York (2008) 1–7
2. The Sensor Web: Distributed Sensing for Collective Action, http://www.sensorsmag.com/networking-communications/the-sensor-web-distributed-sensing-collective-action-1048. Accessed 16 Jul 2010
3. Botts, M., Percivall, G., Reed, C., Davidson, J.: OGC Sensor Web Enablement: Overview and High Level Architecture OGC 06-050r2. Technical Report, Open Geospatial Consortium Inc. (2006)
4. Cox, S.: Observations and Measurements – Part 1 – Observation Schema OGC 07-022r1. Technical report, Open Geospatial Consortium Inc. (2007)
5. Cox, S.: Observations and Measurements – Part 2 – Sampling Features OGC 07-002r3. Technical report, Open Geospatial Consortium Inc. (2007)
6. Botts, M., Robin, A.: OpenGIS Sensor Model Language (SensorML) Implementation Specification OGC 07-000. Technical Report, Open Geospatial Consortium Inc. (2007)
7. Havens, S.: OpenGIS Transducer Markup Language (TML) Implementation Specification OGC 06-010r6. Technical Report, Open Geospatial Consortium Inc. (2006)
8. Na, A., Priest, M.: Sensor Observation Service OGC 06-009r6. Technical Report, Open Geospatial Consortium Inc. (2007)
9. Simonis, I.: OpenGIS Sensor Planning Service Implementation Specification OGC 07-014r3. Technical Report, Open Geospatial Consortium Inc. (2005)
10. Simonis, I., Echterhoff, J.: OGC Sensor Alert Service Implementation Specification OGC 06-028r5. Technical Report, Open Geospatial Consortium Inc. (2007)
11. Simonis, I., Wytzisk, A.: Web Notification Service. Technical Report, Open Geospatial Consortium Inc. (2003)
12. Nebert, D., Whiteside, A., Vretanos, R.: OpenGIS Catalogue Services Specification. Technical Report, Open Geospatial Consortium Inc. (2007)
13. Sheth, A., Henson, C., Sahoo, S.: Semantic sensor web. IEEE Internet Computing 12 (2008) 78–83
14. Chandrasekaran, B., Josephson, J. Benjamins, V.: Ontologies: What are they? Why do we need them? IEEE Intelligent Systems and Their Applications 14 (1999) 20–26

15. Gruber, T.: A translation approach to portable ontologies. Knowledge Acquisition 5 (1993) 199–220
16. Semantic Web for Earth and Environmental Terminology (SWEET) http://sweet.jpl.nasa.gov/sweet/. Accessed 16 Jul 2010
17. Geospatial Ontology, http://geobrain.laits.gmu.edu/ontology. Accessed 16 Jul 2010
18. Russomanno, D., Kothari, C., Thomas, O.: Building a sensor ontology: A practical approach leveraging ISO and OGC models. In: Proceedings of the 2005 International Conference on Artificial Intelligence, CSREA Press, Athens, Georgia, USA (2005) 637–643
19. Schut, P.: OpenGIS Web Processing Service OGC 05-007r7. Technical Report, Open Geospatial Consortium Inc. (2007)
20. Cannataro, M., Comito, C.: A data mining ontology for grid programming. In: Proceedings of the 1st International Workshop on Semantics in Peer-to-Peer and Grid Computing (2003) 113–134
21. Tadepalli, S., Sinha, A.: Ontology driven data mining for geosciences. In: Proceedings of the 2004 AAG Annual Meeting, Association of American Geographers, Washington, D.C., USA (2004 of Conference)
22. Yue, P., Di, L., Yang, W., Yu, G., Zhao, P.: Path planning for chaining geospatial web services. In: Proceedings of the 6th International Symposium on Web and Wireless Geographical Information Systems. Springer, New York (2006) 214–226
23. Morris, R.: A Model for Autonomous Reconfiguration of Sensing Resources. Technical Report, NASA Ames Research Center, Moffett Field, CA USA (2008)
24. Rothermel, R.: A Mathematical Model for Predicting Fire Spread in Wildland Fuels. Technical Report, USDA Forest Service, Washington, D.C., USA (1972)
25. OASIS: OASIS/ebXML Registry Information Model v2.0. Technical Report, OASIS/ebXML Registry Information Model v2.0 (2002)
26. Yue, P., Di, L., Zhao, P., Yang, W., Yu, G., Wei, Y.: Semantic augmentations for geospatial catalogue service. In: Proceedings of the 2006 IEEE International Geoscience and Remote Sensing Symposium, IEEE Press, New York (2006) 3486–3489
27. Yue, P., Di, L., Yang, W., Yu, G., Zhao, P.: Semantics-based automatic composition of geospatial web service chains. Computers & Geosciences 33 (2007) 649–665
28. Whiteside, A., Evans, J.: Web Coverage Service (WCS) Implementation Standard OGC 07-067r5. Technical Report, Open Geospatial Consortium Inc., Wayland, MA, USA (2008)
29. Vretanos, P.: Web Feature Service Implementation Specification OGC 04-094. Technical Report, Open Geospatial Consortium Inc. (2005)
30. Beaujardiere, J.: OGC Web Map Service Interface OGC 03-109r1. Technical Report, Open GIS Consortium Inc. (2004)
31. Zhao, P., Di, L., Yue, P., Yu, G., Yang, W.: Semantic web based geospatial knowledge transformation. Computer & Geosciences 35 (2009) 798–808

Chapter 9
Towards Knowledge-Based Life Science Publication Repositories

Vít Nováček, Tudor Groza, and Siegfried Handschuh

Abstract Despite being a flourishing field, the contemporary online scientific publishing properly exploits mostly raw publication data (rather meaningless bags of words) and shallow meta-data (authors, keywords, citations, etc.) regarding search. The much needed economical mass exploitation of the knowledge implicitly contained in publication texts is still largely an uncharted territory. The way towards filling this gap leads through (1) *extraction* of asserted publication meta-data together with the knowledge implicitly present in the respective text; (2) *integration*, *refinement* and *extension* of the emergent content; (3) *release* of the processed content via a meaning-sensitive search&browse interface catering for services complementary to the current full-text search. This chapter addresses the scientific and engineering challenges related to the suggested approach and introduces a particular solution that tackles them – CORAAL, a prototype for knowledge-based life science publication search.

9.1 Introduction

Digital content processing has no doubt introduced a whole lot of new possibilities of dealing with scientific publications. It makes knowledge much more open and exploitable than in the old "paper times". However, one still needs to go manually through a lot of possibly irrelevant content very often before actually finding the right answers. If we are to make the next step, it is necessary to process knowledge (i.e. concepts and their mutual relations), and not just data or shallow meta-data (i.e. chunks of free text, titles or authors).

Substantial automation of such meaning-intensive information processing is hardly possible with the current industry-strength technologies (e.g. full-text search), since they lack proper support for extraction, representation and processing

V. Nováček (✉)
Digital Enterprise Research Institute (DERI), National University of Ireland, Galway, Ireland
e-mail: vit.nováček@deri.org

H. Chen et al. (eds.), *Semantic e-Science*, Annals of Information Systems 11,
DOI 10.1007/978-1-4419-5908-9_9,

of knowledge implicitly present in texts. As an illustration, imagine for instance finding a support of the claim that *acute granulocytic leukaemia* is different from *T-cell leukaemia*. With the current solutions, it is easy to find articles that contain both or either of the terms; however, the number of results may be quite high (e.g. 593 on PubMed, a state-of-the art life science publication search engine). It is tedious or even impossible to go through all of them in order to find out which of them actually mentions the two leukaemias being different.

To remedy the shortcomings of the current solutions, the future publishing paradigms should support decomposed machine-readable content that goes beyond mere text locked in rather monolithic publications. The envisioned content should allow for expressive inter-linking of scientific artefacts, thus unfolding new dimensions of browsing and data integration. It should be amenable to robust autonomous extraction and meaningful inference of implicit domain knowledge present in the text in order to expose it for search, too.

9.1.1 Motivation

However, a scalable acquisition of expressive machine-readable knowledge from unstructured resources is a major challenge [1–3], mainly due to the fact that the manual knowledge acquisition is tedious, expensive and error-prone in most practical settings.

Ontology learning [4] from text, possibly combined with the emergent semantics principles [5, 6], is therefore often considered as a viable step towards tackling the knowledge acquisition bottleneck. The automatically extracted emergent knowledge is dynamic, often explicitly vague [7, 8], potentially inconsistent and incorrect, though.

Approaches handling these features within traditional logics-based KR&R have been proposed in the literature [9–12]. However, a practical, robust and general-purpose modelling covering the emergent knowledge is deemed to be hardly possible using these paradigms [13]. The main reasons for that are severe theoretical challenges, intractability, restricted applicability and low accessibility for untrained users. Moreover, there are substantial conceptual peculiarities hampering using the prevalent logical KR&R for the emergent knowledge. The shallow structure of learned ontologies does not typically allow for many non-trivial logical conclusions. Potential incorrectness of the emergent facts (i.e. empirical nature of truth) is awkward to be modelled by any logical knowledge representation (which by definition requires a categorical notion of truth, or at least its measure or degree in the uncertain generalisations of logics).

In addition, recent advances in neuroscience [14] show that logical reasoning as such is not really used by the most robust inference engines we know of – people – in order to cope with noisy and incomplete information efficiently. References [14] and [15] suggest that a similarity-based inference is more appropriate (although possibly incomplete and unsound) in practical settings involving noisy real-world data,

while the logical reasoning performs in a satisfactory manner only when sufficiently formalised knowledge is a priori available.

Building on the discussed motivations, we have researched a light-weight similarity-based framework for coping with the noisy and sparse emergent knowledge. The framework is essentially based on a simple formal notion of conceptual similarities. It allows for soft incremental refinement and/or extension of the emergent content and for approximate querying of the resulting knowledge bases. The framework can be exploited in a simple and intuitive way, which is a very important feature given the life sciences use case and community we primarily target.

On the top of our novel approach to emergent knowledge processing, we have implemented CORAAL[1] a proof-of-concept prototype for semantic life science publication search. It supports not only mere keyword queries but also search for expressive statements and knowledge-based faceted browsing of the respective results. CORAAL is essentially a first step towards a powerful vehicle of future e-Science – semantic publication repositories being built from the vast amounts of legacy data available in scientific articles and exposing the knowledge scattered within them in unprecedented scale and depth.

9.1.2 State-of-the-Art Overview

Since the need for more efficient exploitation of life science publications emerged from the respective community quite some time ago, there are already systems targeting the problems we aim to tackle. Figure 9.1 displays the most relevant applications in the life sciences domain in a plot with two axes – *effort* and *benefit* (the placement is only orientational, though, as it does not reflect any formal measure related to the particular systems). The *effort* axis indicates how much more or less

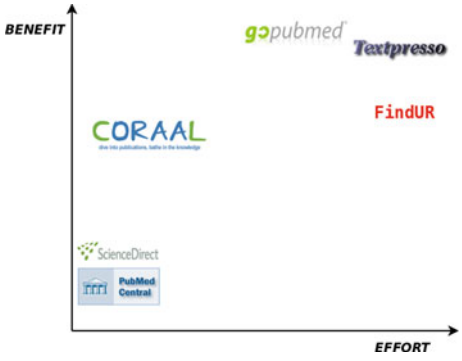


Fig. 9.1 Informative comparison of selected systems

[1]CORAAL stands for *COntent extended by emeRgent and Asserted Annotations of Linked publication data*.

manual effort must the creators and/or maintainers of a tool spend before it can perform sufficiently, or before it can be ported to a new domain. The *benefit* axis reflects how much benefit users get when searching for the knowledge hidden in publications with a tool.

The state-of-the-art applications like ScienceDirect or PubMed Central require almost no effort in order to expose arbitrary life science publications for search (therefore we used them as a base-line in the user-centric experiment). However, the benefit they provide is rather limited when compared to cutting-edge approaches aimed at utilising also the publication knowledge within the query construction and/or result visualisation. Such innovative solutions may require much more a priori effort in order to work properly, though.

FindUR [16], Melisa [17] and GoPubMed [18] are ontology-based front-ends to a traditional publication full-text search. They allow either for effective restriction and intelligent visualisation of the query results (GoPubMed) or for focusing the queries onto particular topics based on an ontology (FindUR and Melisa). FindUR and Melisa use a Description Logics [19] ontology built from scratch and a custom ontology based on MeSH (cf. http://www.nlm.nih.gov/mesh/), respectively. GoPubMed dynamically extracts parts of the Gene Ontology (cf. http://www.geneontology.org/) relevant to the query, which are then used for restriction and a sophisticated visualisation of the classical PubMed search results. None of the tools, nevertheless, offers querying for or browsing of arbitrary publication knowledge – terms and relations not present in the systems' rather static ontologies simply cannot be reflected in the search.

Textpresso [20] enables searching for relations between concepts in particular chunks of text (namely for gene-to-gene interactions). However, the underlying ontologies and their instance sets have to be provided manually. Moreover, the system's scale regarding the number of publications' full-texts and concepts covered is quite low.

9.1.3 Main Contributions and Structure of the Chapter

From the overview of the state of the art in the field, it is obvious that the biggest challenge is a reliable automation of more expressive content acquisition. None of the related systems addresses this problem appropriately, which makes them either poorly scalable or difficult to port to a new domain. The economical aspect related to the manual population, maintenance and curation of the data is also important – the respective costs of human resources may be quite high.

In the following, we introduce CORAAL, a system that substantially reduces the manual efforts needed to create a knowledge-based information system. An overview of our solution is given in Section 9.2. Section 9.3 presents details of the novel emergent knowledge processing framework that supports the necessary automated text analysis. The particular way of how we process publications in CORAAL is described in Section 9.4. We comment on typical usage of the actual tool in Section 9.5. Section 9.6 discusses indicative results of CORAAL tests with real

users. Summary of related work is given in Section 9.7. We conclude the chapter and present our future work in Section 9.8.

9.2 Overview of Our Approach

In order to support comprehensive search functionalities, we propose to complement a standard (full-text) publication search approach with advanced services catering for semantic search. By semantic search we mean querying for and browsing of expressive statements capturing relations between concepts in the respective source articles.

Our particular solution – the CORAAL prototype – is built on top of the KONNEX [21, 22] framework and the EUREEKA library that implement the principles introduced in Section 9.3. It runs in a client–server mode. In order to work with the tool, you only need your web browser. Everything else is handled by the server, quite similarly to the classical search engines (e.g. Google) from the user's point of view. The technical architecture of CORAAL is depicted in Fig. 9.2. EUREEKA provides for knowledge extraction from text and other knowledge resources (e.g. ontologies or machine-readable thesauri like NCI and EMTREE) via the *knowledge*

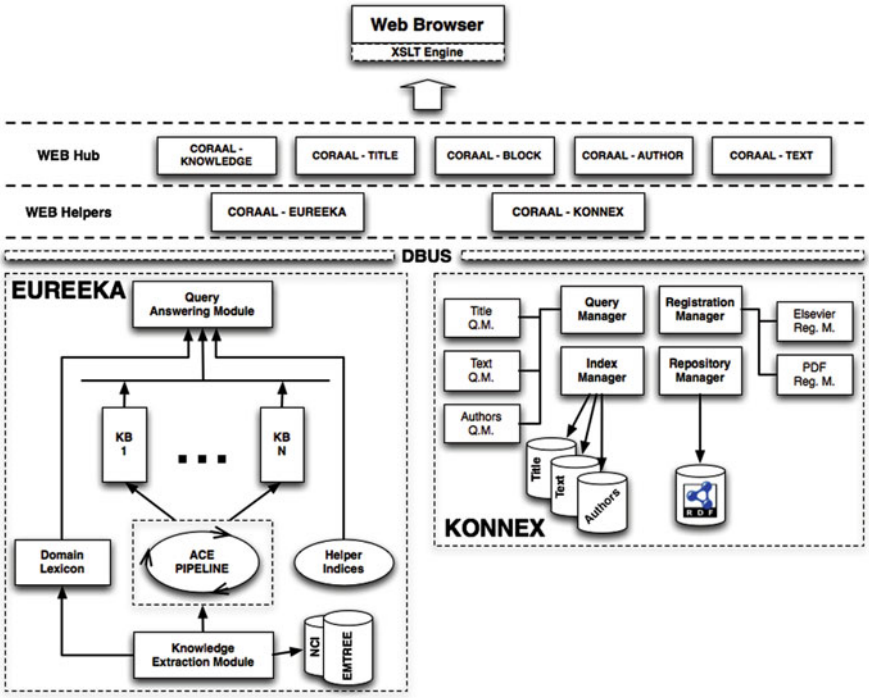


Fig. 9.2 Architecture of our solution

extraction module. The extraction process possibly updates the *domain lexicon* and produces new knowledge being processed in the *ACE pipeline*[2] before addition into particular *knowledge bases*, which may be multiple if we want to represent particular contexts of the domain of interest separately. Details of the work-flow are elaborated in Section 9.4.2. The knowledge bases are exposed to consumers via a semantic *query answering module*, optimising the retrieval and sorting the results using *helper indices* on the stored data. KONNEX tackles the integration of the extracted publication text and meta-data (in the form of RDF graphs) in a triple store. Operations related to data *registration* (inclusion and integration with the stored content), *repository* maintenance, full-text *query* processing and *indices* are handled by respective *manager* modules, possibly composed of sub-modules handling particular data or query types.

There are several conceptually separate modules in CORAAL; moreover, EUREEKA is written in the Python programming language, while KONNEX is in Java. Therefore we utilise an inter-process communication layer implemented using the D-BUS framework (cf. http://en.wikipedia.org/wiki/D-Bus). On the top of the core-level EUREEKA and KONNEX APIs, a set of helper web services rests. These manage the user requests and forward the data returned by the core APIs to the web hub, which is a set of Java servlets handling particular types of search. The servlets produce machine-readable RDF representing answers to user queries. The RDF has XSL stylesheets attached in order to render the results in a human-readable form by the web browser itself via the Exhibit front-end.[3] This solution results in CORAAL being a pure Semantic Web [23] application, as the data-flow between the core infrastructure and the other modules is strictly based on RDF graphs. While being presented in a human-readable form in the browser, the produced data can be directly analysed by an application or fetched by a crawler as well.

9.3 Emergent Knowledge Processing Framework

In this part we describe the novel technology that powers CORAAL. First, the proposed empirical knowledge representation principles are tailored to emergent data (Section 9.3.1). Respective inference services are summarised in Section 9.3.2. Our approach to an efficient implementation of a respective framework is outlined in Section 9.3.3 then.

9.3.1 Empirical Knowledge Representation

As can be observed for instance in [7], the output of automatic knowledge extraction (i.e. ontology learning) techniques can mostly be reduced to various types of binary relations between sets of lexical entities. These relations may possibly occur in a

[2] ACE stands for Addition, Closure, Extension. See Section 9.4.2 for details.

[3] Cf. http://www.simile-widgets.org/exhibit/. Details on how to use the CORAAL user interface are given in Section 9.5.

negative form (e.g. difference or relations extracted from negative grammatical constructions). Finally, the ontology learning results usually come with a heuristically computed confidence measure. A respective compact representation of emergent concepts and knowledge bases is given by the following definition.

Definition 1 *Concept* is a square matrix $\mathbf{A}$ with elements $a_{i,j} \in [-1, 1]$, $i,j \in I$, where I is an index set. Let M be a set of all concepts, L a set of lexical expressions that may refer to concepts in M and L^* a set of fuzzy sets [24] defined on the L universe. We define *lexical interpretation* λ as a bijection $\lambda : M \rightarrow L^*$. *Index assignment* binding the concepts and indices together is then a bijection $ind : M \rightarrow I$ such that $ind(\mathbf{A}) = ind(\mathbf{B})$ iff $\lambda(\mathbf{A}) = \lambda(\mathbf{B})$. Regarding concept equivalence, we call concepts *strongly equal*, $\mathbf{A} = \mathbf{B}$, iff $a_{i,j} = b_{i,j}$ for $\forall i,j \in I$, and *weakly equal*, $\mathbf{A} \simeq \mathbf{B}$, iff $ind(\mathbf{A}) = ind(\mathbf{B})$. *Empirical knowledge base* is a tuple $(K, I_K, L_K, ind_K, \lambda_K)$, where $K \subseteq M, I_K \subseteq I, L_K \subseteq L$ and ind_K, λ_K are the respective specific index assignment and lexical interpretation mappings.

Note that we do not distinguish between "classes" and "individuals" in the traditional sense (i.e. sets and elements in a domain universe, respectively). A concept can be empirically considered to be an "individual" as long as it has no sub-types, but it can suddenly become a "class" when a sub-type concept is newly introduced to it. Therefore everything is just a concept and finer-grained ontological distinctions are left to particular applications of the basic principles.

The sets M, L^*, I can be understood as the conceptual, symbolic and real-world domain, respectively, in the semiotic triangle [25] perspective (considering I as a set of unique identifiers of entities existing in universe). The λ, ind mappings (and their inverses) can then be understood along the symbolisation and reference relations in the triangle. The intuition behind the fuzzy character of λ is the fact that concepts are usually referred to by more than one lexical expression. Moreover, these expressions have uneven degrees of relevance w.r.t. the particular concept (e.g. the expression *"a reasoning erected primate"* is perhaps a bit more relevant to the *"human"* concept than the expression *"a bipedal animal without feathers"*, while the *"human"* expression is one of the most relevant).

The introduced matrix notation for concepts is convenient due to its conciseness; however, we can use also a more human-readable and explanatory *statement notation*, closely following the standard RDF(S) terminology [26]. A concept $\mathbf{A}$ can be expanded as a conjunction

$$< s : p_1 : o_1 >^{a_{p_1,o_1}} AND < s : p_2 : o_2 >^{a_{p_2,o_2}} AND \ldots AND < s : p_n : o_n >^{a_{p_n,o_n}}$$

of particular *subject : predicate : object* statements.[4] $s = \text{ind}(\mathbf{A})$ and n is the number of non-zero elements in $\mathbf{A}$. p_i, o_i are the row and column indices, respectively, of the particular non-zero matrix element. The element values a_{p_i,o_i} represent the degrees

[4]Note that without loss of generality, URIs may serve as concept indices in the statements. Consequently, ind^{-1} de facto plays a role of the URI dereference. To facilitate readability, we provide simply lexical terms instead of indices or URIs in the examples throughout the chapter, though.

of certainty about the fact that the actual relation $\text{ind}^{-1}(p_i)$ holds (or does not hold for $a_{p_i,o_i} < 0$) between $\text{ind}^{-1}(s)$ and $\text{ind}^{-1}(o_i)$.

Example 1 Consider for instance the concept `T-cell leukemia`, being certainly a `type of` the concept `disease` and certainly *not* a `type of` (i.e. different from) the concept `infection` according to a human expert. The respective concept matrix **A** may look like this (omitting the zero elements):

SAMPLE-URI#1	SAMPLE-URI#3	SAMPLE-URI#4
SAMPLE-URI#2	1.0	-1.0

The corresponding statement notation would be

`<SAMPLE-URI#1 : SAMPLE-URI#2 : SAMPLE-URI#3>`$^{1.0}$ *AND* `<SAMPLE-URI#1 : SAMPLE-URI#2 : SAMPLE-URI#4>`$^{-1.0}$

The `SAMPLE-URI#2`, `SAMPLE-URI#3` and `SAMPLE-URI#4` indices correspond to matrices **B**, **C** and **D**, respectively, regarding the ind mapping. The lexical interpretation λ is defined as follows then: $\lambda(\mathbf{A}) = (S_1, \mu_1), \lambda(\mathbf{B}) = (S_2, \mu_2)$, $\lambda(\mathbf{C}) = (S_3, \mu_3), \lambda(\mathbf{D}) = (S_4, \mu_4)$, where $(S_1, \mu_1), (S_2, \mu_2), (S_3, \mu_3)$ and (S_4, μ_4) are fuzzy sets such that μ_1(`T-cell leukemia`) $= 1$, μ_2(`type of`) $= 1$, μ_3(`disease`) $= 1$, μ_4(`infection`) $= 1$.

The interpretation of the element values in the concept matrices is deliberately left unrestricted, since the degrees are meant to capture heuristic confidence bound to emergent statements. The confidence may be based on statistics; however, it may also be based on arbitrary measures assigned either by people or by various knowledge extraction algorithms, therefore no particular mathematical formalism can be used for the degree interpretation in general. Users can, however, impose a specific semantics on degrees later on when consuming the data stored in an empirical knowledge base. The most straightforward approach is interpreting the absolute values bound to the positive and negative facts as membership degrees, following the fuzzy sets formalism [24]. Intuitionistic fuzzy sets [27] can be used for more straightforward interpretation of positive and negative degrees as membership and non-membership measures, respectively.

Concepts stored in a knowledge base can arbitrarily change in the emergent settings, which is supported by the following operation (note that the possible updates of the ind_K, λ_K mappings are technical issues dependent on the particular implementation and thus omitted in the definition).

Definition 2 *Concept change* $\Delta_{u,v} : M^2 \to M, u, v \in [0, 1]$ is a parametrised binary operation such that $\Delta_{u,v}(\mathbf{A}, \mathbf{B}) = \mathbf{C}$, where $c_{i,j} = \nu_{u,v}(a_{i,j}, b_{i,j})$. $\nu_{u,v} : [-1, 1]^2 \to [-1, 1]$ is a so-called *element change* function defined as follows: (1) for $x \neq 0$, $\nu_{u,v}(x, 0) = x$; (2) for $y \neq 0$, $\nu_{u,v}(0, y) = y$; (3) for $x, y \neq 0$, $\nu_{u,v}(x, y) = F(ux, vy)$, F being an ordered weighted averaging (OWA) operator.[5]

[5]Defined in [28] as $F(a_1, \ldots, a_n) = \sum_{j=1}^{n} w_j b_j$, where b_j is the jth largest of the a_i and w_j are a collection of weights (also called a weight vector) such that $w_j \in [0, 1]$ and $\sum_{j=1}^{n} w_j = 1$. Note that we use the additional u, v weights in order to explicitly capture the relative relevance of the $\Delta_{u,v}$ first and second argument independently from their relative sizes.

The change operation can be understood as a simple, yet useful formal model of cognitive learning and attitude change as studied in psychology [29], i.e. acceptance, rejection or modification of the attitude on a topic – a relation between two concepts in our case – according to the current content of the knowledge base and persuasive emergent facts. The purpose of the *u, v* parameters is to reflect the "persuasion potential" (weight) of particular knowledge sources being incorporated w.r.t. the already known content. For instance, they may be set as $u = 1, v = 1$ for presumably correct and equally trusted knowledge from manually created ontologies, $u = 0.9, v = 1$ for input knowledge from a resource more trusted than the actual content, or $u = 1, v = 0.75$ for less trusted learned data.

Using an appropriate selection of the F operator (see [28] for details on possible choices), one can model various types of concept change. The following are the most prominent special cases: (i) F as the min function – gives strict preference to the negative or less certain facts, which is perhaps not very practical in most realistic settings; (ii) F as the max function – gives strict preference to the positive or more certain facts, which may be practical when negative knowledge is rather sparse and/or less trusted among the input data; (iii) F as the arithmetic mean function – aggregating both current and new values, possibly adjusting their relative relevance according to *u, v*, though; (iv) F with a dynamic weight vector derived according to the particular u, v, x, y values[6] to model weighted arithmetic mean – similar to the previous case, however, resulting in the actual $v_{u,v} = \frac{ux+vy}{u+v}$ function instead of $v_{u,v} = \frac{ux+vy}{2}$.

Example 2 Imagine that we learn from a natural language text that `T-cell leukemia` is different from `acute granulocytic leukemia` with confidence 0.8, and that `T-cell leukemia` is a type of `infection` with confidence 0.2. Assuming the 0.8 relevance for the learning algorithm when compared to the 1.0 relevance of human expert, the `T-cell leukemia` concept from Example 1 has to be updated using $\Delta_{1.0,0.8}$ regarding the new findings. The changed concept is described as follows then (degrees computed using the dynamic weighted mean OWA operator):

```
<T-cell leukemia : type : disease>1.0 AND <T-cell leukemia : type : acute
granulocytic leukemia>-0.8 AND <T-cell leukemia : type : infection>-0.46
```

Besides direct concept incorporation by the change operation, one has to be able to aggregate multiple concepts evenly as well. This is particularly useful for instance when merging concepts from multiple equally trusted sources before their actual incorporation into the known content or when aggregating intermediate inference results.

Definition 3 *Concept aggregation* is a function $\bigcirc : 2^M \to 2^M$, $\bigcirc(X) = \{\odot(S_l) | S_l \in S\}$, where S is a set of the equivalence classes on X *w.r.t.* $\simeq$. $\odot : 2^M \to M$ is a function aggregating weakly equal matrices $\mathbf{A}_1, \ldots, \mathbf{A}_n$ into a matrix $\mathbf{B}$ with

[6] Essentially only one value w fully dependent on u, v, x, y is to be derived, since the remaining element of the OWA weight vector of size 2 is equal to $1 - w$.

elements $b_{i,j} = F(x_1, \ldots, x_k)$, where $x_1, \ldots, x_k$ are the respective non-zero $a_{i,j}$ values among $\mathbf{A}_1, \ldots, \mathbf{A}_n$ and F is an OWA operator.

The function essentially merges the weakly equal input concepts into one matrix without explicitly prioritising any argument. The actual F operator selection results in various aggregation characteristics, most notably the following:

(i) F as the max function – either the most trusted positive or the least trusted negative degrees prevail, leading to the general preference of positive knowledge;
(ii) F as the min function – either the most trusted negative or the least trusted positive degrees prevail, leading to the general preference of negative knowledge;
(iii) F as the arithmetic mean function – an aggregation treating both positive and negative knowledge evenly.

Example 3 An aggregation of the `T-cell leukemia` concept updated in Example 2 with the following concept of the same relevance (e.g. learned, too, but from different data)

```
<T-cell leukemia : type : acute granulocytic leukemia>^-0.5 AND <T-cell
leukemia : type : infection>^-0.8
```

would result in this update (degrees computed using arithmetic mean OWA operator):

```
<T-cell leukemia : type : disease>^1.0 AND <T-cell leukemia : type : acute
granulocytic leukemia>^-0.65 AND <T-cell leukemia : type : infection>^-0.63
```

Crucial for the basic inference services in our approach is the notion of similarity. We formalise it using metrics on M.

Definition 4 Semantic metrics class Ω is a set of parametrised functions $\delta_H : M^2 \to \mathbb{R}_0^+$ *for all* $\emptyset \subset H \subseteq I^2$ such that: (1) δH is a metric on M; (2) in order to compute $\delta_H(\mathbf{A}, \mathbf{B})$, only elements $a_{i,j}, b_{i,j}$ with $(i,j) \in H$ are taken into account. We can define a partial ordering $\preceq$ o$n\,\Omega$, such that $\delta_{H_1} \preceq \delta_{H_2}$ iff $H_1 \subseteq H_2$. Dually to distance, we define graded *concept similarity*[7] as $\sigma_H : M^2 \to (0, 1], \sigma_H(\mathbf{A}, \mathbf{B}) = \frac{1}{1+\delta_H(\mathbf{A}, \mathbf{B})}$. A partial ordering $\sqsubseteq$ on the set of all similarities can be defined as $\sigma_{H_1} \sqsubseteq \sigma_{H_2}$ *iff* $\delta_{H_1} \preceq \delta_{H_2}$.

An example metric is $\delta_H(\mathbf{A}, \mathbf{B}) = \sum_{(i,j)\in H} |a_{i,j} - b_{i,j}|$, or a normalised alternative $\delta_H(\mathbf{A}, \mathbf{B}) = \frac{1}{|H|} \sum_{(i,j)\in H} |a_{i,j} - b_{i,j}|$. The dependence on the H set allows for graded modelling of specific distances, influenced only by certain relations instead of all relations possible. The specificness of the particular distances (or the dual similarities) is directly related to the $\leq$ ordering, i.e. if $\delta_{H_1} \preceq \delta_{H_2}$, then δ_{H_1} is more specific

[7] The duality w.r.t. the distance is ensured by the conformance to two intuitive conditions – inverse proportionality and equality to 1 when the distance is 0.

than δ_{H_2}. Specific similarities are particularly useful when we want to retrieve content from a knowledge base – e.g. all concepts being type of a disease and treated by radiological methods. We can form a respective query concept and check the knowledge base for matrices with specific similarity regarding the two query properties higher than a given threshold. Comparison regarding all possible properties would possibly retrieve much smaller set of appropriate answer concepts for large knowledge bases with many properties present, which is not the intuitively expected behaviour.

We can distinguish certain prominent types of similarity functions according to the H parameter. First, let $\sigma(\mathbf{A},\mathbf{B}) = \sigma_{I^2}(\mathbf{A},\mathbf{B})$ be a similarity *between* $\mathbf{A}$ and $\mathbf{B}$ (a general comparison). Second, let $\overleftarrow{\sigma}(\mathbf{A},\mathbf{B}) = \sigma_{\{(i,j)|b_{i,j}\neq 0\}}(\mathbf{A},\mathbf{B})$ and $\overrightarrow{\sigma}(\mathbf{A},\mathbf{B}) = \sigma_{\{(i,j)|a_{i,j}\neq 0\}}(\mathbf{A},\mathbf{B})$ be a similarity of $\mathbf{B}$ *to* $\mathbf{A}$ and $\mathbf{A}$ *to* $\mathbf{B}$, respectively (a specific comparison of either $\mathbf{B}$ to $\mathbf{A}$, or $\mathbf{A}$ to $\mathbf{B}$, based on the respective non-zero elements). Third, let $\overline{\sigma}(\mathbf{A},\mathbf{B}) = \sigma_{\{(i,j)|a_{i,j}\neq 0 \wedge b_{i,j}\neq 0\}}(\mathbf{A},\mathbf{B})$ be an intersection similarity between $\mathbf{A}$ and $\mathbf{B}$ (a comparison based only on the elements $\mathbf{A}$ and $\mathbf{B}$ have in common). Quite clearly, $\overline{\sigma} \sqsubseteq \overrightarrow{\sigma} \sqsubseteq \sigma, \overline{\sigma} \sqsubseteq \overleftarrow{\sigma} \sqsubseteq \sigma$.

Example 4 The similarity $\overrightarrow{\sigma}$ of the following concept (with the subject indicating a variable)

```
<?X : type : disease>^1.0 AND <?X : type : acute granulocytic leukemia>^-1.0
```

to the concept

```
<T-cell leukemia : type : disease>^1.0 AND <T-cell leukemia : type : acute
granulocytic leukemia>^-0.65 AND <T-cell leukemia : type : infection>^-0.63
```

is about 0.851 when using the $\delta_H(\mathbf{A},\mathbf{B}) = \frac{1}{|H|}\sum_{(i,j)\in H}|a_{i,j} - b_{i,j}|$ distance as a basis for the similarity computation. The respective σ similarity *between* the concepts is about 0.753 then. Note that both similarities are relatively high, suggesting that `T-cell leukemia` might be an instance of `?X` to a certain degree.

The presented knowledge representation follows the open world assumption by default (see for instance [30, 31] for discussion of open and closed world assumptions). The zero degrees in the concept matrices do not mean that the respective statements are false, they only indicate that currently nothing is known about them. In order to simulate closed world behaviour if needed, we can use the following operator that essentially makes the unknown knowledge explicitly negative.

Definition 5 *Closed world operator* is a parametrised function $\Theta_G : M \rightarrow M$, where $G \subseteq I^2$. $\Theta_G(\mathbf{A}) = \mathbf{B}$ such that $b_{i,j} = a_{i,j}$ *if* $a_{i,j} \neq 0$*;* $b_{i,j} = -1$ *if* $a_{i,j} = 0 \wedge (i,j) \in G$.

Since the σ_H similarity is the basis of every inference operation, Θ_G is to be applied namely in the similarity computations. The G parameter is derived from the non-zero indices of the arguments then, i.e. $\sigma_H(\mathbf{A},\mathbf{B})$ is actually computed as $\sigma_H(\Theta_G(\mathbf{A}),\Theta_G(\mathbf{B}))$ under the closed world assumption, where $G = \{(i,j)|a_{i,j} \neq 0\} \cup \{(k,l)|b_{k,l} \neq 0\}$.

Example 5 Under the closed world assumption, the first concept from Example 4 changes to

<?X : is a : disease>$^{1.0}$ *AND* <?X : is a : acute granulocytic leukemia>$^{-1.0}$ *AND* <T-cell leukemia : is a : infection>$^{-1.0}$

regarding the similarity computation, while the second one remains unchanged. The similarities $\overrightarrow{\sigma}$ and σ are obviously the same under the closed world assumption and equal to about 0.807.

9.3.2 Inference Services

In our conception of reasoning, we share the general standpoint to automated inference discussed in cognitive, memory- and similarity-based approaches (see for instance [32, 33]). However, all the proposed services are tailored to the relatively simple structure of the emergent knowledge and thus are rather light-weight, building on the notion of graded similarities presented in Definition 4.

The very basic inference service is a *retrieval of similar concepts*. Given a knowledge base $(K, I_K, L_K, ind_K, \lambda_K)$, the retrieval is a parametrised function $\alpha_\epsilon^K : M \rightarrow 2^M$ defined as follows:

$$\alpha_\epsilon^K(\mathbf{Q}) = \{\overleftarrow{\sigma}(\mathbf{A}, \mathbf{Q})\mathbf{A} | \mathbf{A} \in K \wedge \overleftarrow{\sigma}(\mathbf{A}, \mathbf{Q}) \geq \epsilon\},$$

where $\epsilon \in [0, 1]$ is a threshold. The matrix indices in the retrieved set may be kept as present in K or changed to a single artificial index if needed. The retrieval essentially checks the knowledge base for approximate satisfiability of the conjunction of statements that form the matrix $\mathbf{Q}$. Then it returns the respective concepts weighed by the actual similarity to $\mathbf{Q}$.

Example 6 Let us take the concept

<?X : is a : disease>$^{1.0}$ *AND* <?X : is a : acute granulocytic leukemia>$^{-1.0}$

from Example 4 as a query $\mathbf{Q}$ on K consisting of the following single concept $\mathbf{A}$:

<T-cell leukemia : is a : disease>$^{1.0}$ *AND* <T-cell leukemia : is a : acute granulocytic leukemia>$^{-0.65}$ *AND* <T-cell leukemia : is & : infection>$^{-0.6\bar{3}}$

The similarity $\overrightarrow{\sigma}(\mathbf{Q}, \mathbf{A}) = \overleftarrow{\sigma}(\mathbf{A}, \mathbf{Q}) \approx 0.851$ when using the $\frac{1}{|H|}\sum_{(i,j)\in H} |a_{i,j} - b_{i,j}|$ distance. The result of $\alpha_\epsilon^K(\mathbf{Q})$, assuming $\epsilon \leq \overleftarrow{\sigma}(\mathbf{A}, \mathbf{Q})$, is approximately

<T-cell leukemia : is a : disease>$^{0.851}$ *AND* <T-cell leukemia : is a : acute granulocytic leukemia>$^{-0.553}$ *AND* <T-cell leukemia : is a : infection>$^{-0.539}$

Building on the similar concepts retrieval, we coin a simple *analogical extension of concepts*, which is a parametrised function $\tau_\epsilon^K : M \rightarrow M$ defined as follows:

$$\tau_\epsilon^K(\mathbf{T}) = \mathbf{T} + pop(\bigcirc(\{\overline{\mathbf{X}} | \mathbf{X} \in \alpha_\epsilon^K(\mathbf{T})\})),$$

where *pop* is a function selecting an arbitrary element from a set and $\overline{\mathbf{X}}$ is a matrix with elements $\bar{x}_{i,j} = x_{i,j}$ if $t_{i,j} = 0 \wedge x_{i,j} \neq 0$, $\bar{x}_{i,j} = 0$ otherwise. The source elements of the $\alpha_{\epsilon}^{K}(\mathbf{T})$ set are supposed to have the same index as the $\mathbf{T}$ target concept in this case (thus *pop*() is deterministic, since the $\bigcirc$ result is a singleton). Applying an extension typically makes sense when we have a target concept we do not know much about and assume presence of similar concepts in the knowledge base $(K, I_K, L_K, \text{ind}_K, \lambda_K)$ that are relatively well-specified and therefore appropriate for an extension of the target.

Example 7 The extension of the target concept

<?X : is a : disease>$^{1.0}$ *AND* <?X : is a : acute granulocytic leukemia>$^{-1.0}$

from Examples 4 and 6 by the source concept

<T-cell leukemia : is a : disease>$^{1.0}$ *AND* <T-cell leukemia : is a : acute granulocytic leukemia>$^{-0.65}$ *AND* <T-cell leukemia : is a : in- fection>$^{-0.63}$

is approximately

<T-cell leukemia : is a : disease>$^{1.0}$ *AND* <T-cell leukemia : is a : acute granulocytic leukemia>$^{-1.0}$ *AND* <T-cell leukemia : is a : in- fection>$^{-0.539}$

when using α_{ϵ}^{K} in the τ_{ϵ}^{K} computation as used in Example 6.

To enable *custom semantics specification* and expressive *query-answering*, we define simple, yet already quite powerful conjunctive rules and queries with negation.

Definition 6 Rule is composed of antecedent and consequent statement sets, bound in a conditional expression with a weight *w:*

$$< s_1 : p_1 : o_1 >^{d_1} AND \ldots AND < s_i : p_i : o_i >^{d_i \leftarrow w} < s_{i+1} : p_{i+1} : o_{i+1} >^{d_{i+1}}$$
$$AND \ldots AND < s_n : p_n : o_n >^{d_n}$$

A brief rule notation $(\widehat{A}, \widehat{C}, w)$ compresses the antecedent and consequent statements into the respective $\widehat{A}, \widehat{C}$ sets of concept matrices. *Query* is a rule with no associated weight and an empty consequent set.

The understanding of rules is quite usual – if the antecedent statements are satisfied by a data set (i.e. an empirical knowledge base), the consequent statements are added to the data set, taking the weight of the rule into account within the addition process. The particular notions of statement *satisfaction* and *addition* are based on the similarities and concept change operator defined in Section 9.3.1. Queries are very similar to rules then – if the query statements are satisfied by a set of concepts, the concepts are returned, taking the actual similarity of the query to the data set into account, though.

Details on rule evaluation are given in Algorithms 1–3 below. The presented algorithms operate on rules (and queries) satisfying the following additional conditions:

1. Each antecedent concept contains at most one variable in the object position.
2. For every variable *?R* occurring in the predicate position in any statement, there must be an antecedent **A** with the *?R* subject and no variables in predicate positions; if there is a variable present as an object in **A**, it must occur as a subject in no antecedent.

For such rules, we can instantiate possible predicate variables first[8] and thus Algorithms 1–3 can work with an expanded set of rules containing only definite concepts as predicates.

Materialisation of a rule $\mathcal{R}$ w.r.t. a knowledge base $(K, I_K, L_K, \text{ind}_K, \lambda_K)$ is computed by the *applyRule*() function given in Algorithm 1. Note that query answering essentially means calling the *applyRule*() function only, while a materialisation of a rule set w.r.t. a knowledge base requires further processing by Algorithms 2 and 3.

Algorithm 1 The $applyRule()$ function

Require: $\mathcal{R}, v, C, V, K, \epsilon$ – the input rule, variable, its instance candidates, variable to instance mapping, concept set and a threshold, respectively
Require: the $testTermination()$, $getNext()$ functions
$applied \leftarrow \emptyset$
for $c \in C$ **do**
  $V[v] \leftarrow c$
  $l \leftarrow testTermination(\mathcal{R}, V, K, \epsilon)$
  if $l \neq \emptyset$ **then**
    $applied \leftarrow applied \cup l$
    continue
  else
    $\overline{v}, \overline{C} \leftarrow getNext(\mathcal{R}, v, V, K)$
    if $\overline{v} =$ **nil OR** $\overline{C} =$ **nil then**
      continue
    end if
    $applied \leftarrow applied \cup applyRule(\mathcal{R}, \overline{v}, \overline{C}, V, K, \epsilon)$
  end if
end for
return *applied*

The *getNext*() function chooses the next variable and respective instance candidates to try out (both can possibly be empty, i.e. *nil*). The next variable is the object variable of the $\mathcal{R}$ antecedent corresponding to *v*, while the candidates are respective objects retrieved from *c*.

The *testTermination*() function checks if all variables in *v* are bound to instances and if the respective instantiated statements from the $\widehat{A}$ antecedents of $\mathcal{R} = (\widehat{A}, \widehat{C}, w)$ are existent in *K*. If not, the function returns ø. Otherwise, it first computes $F(S)$, where F is an OWA operator and S is a vector of similarities $\overrightarrow{\sigma}(\mathbf{A}, \mathbf{B})$ regarding all instantiated matrices $\mathbf{A} \in \widehat{A}$ and the respective $\mathbf{B} \in K$ matrices such

[8] By iterating through a respective knowledge base and/or by similar concept retrieval.

that $\mathbf{B} \sim \mathbf{A}$. If $F(S) \geq \epsilon$, a set of $\Delta_{1-F(S),w}(\mathbf{C}_K, \mathbf{C})$ elements is returned for rules, where $\mathbf{C}_K, \mathbf{C}$ range through K and instantiated consequent matrices from $\widehat{C}$ such that $\mathbf{C}_K \simeq \mathbf{C}$. For queries, the *testTermination*() function returns simply concepts $F(S)\mathbf{A}_K$, where $\mathbf{A}_K$ are all concepts from K such that $\mathbf{A}_K \simeq \mathbf{A}$, ranging through all instantiated matrices $\mathbf{A}$ corresponding to the $\widehat{A}$ antecedent set.

Three choices of the F operator are quite straightforward in the *testTermination*() function – either maximum, or average, or minimum. Maximum is the least appropriate, though, since it can cause a rule to trigger even in case when one antecedent matrix is sufficiently similar to the content of the knowledge base, while the others are rather dissimilar. Average and minimum functions result in more desirable behaviour (minimum being more strict than average).

The intuition behind setting the parameters for the $\Delta_{u,v}$ concept change application in *testTermination*() is as follows. The relevance of the current knowledge base content w.r.t. rule consequents being incorporated should be inversely proportional to the aggregated similarity of the antecedents (i.e., the less closely the antecedents are matched, the lower importance the consequents have in case of a conflict). For queries, we merely retrieve concepts from the knowledge base following the antecedent set instantiation. The retrieved concepts are, however, weighed by the aggregate similarity to reflect the degree of the actual match.

Example 8 Let us consider K consisting of two concepts $\mathbf{A}$:

```
<T-cell leukemia : is a : non-Hodgkin's lymphoma>^0.8 AND
<T-cell le-ukemia : is a : leukemia>^0.8
```

and $\mathbf{B}$:

```
<non-Hodgkin's lymphoma : is a : leukemia>^1.0
```

Application of the rule `<?X : isA : ?Y>`^{1} *AND* `<?Y : isA : ?Z>`$^1 \rightarrow^1$ `<?X : isA : ?Z>`^{1} on K results in `?X, ?Y, ?Z` being instantiated to `T-cell leukemia, non-Hodgkin's lymphoma, leukemia`, respectively. The instantiated antecedent matrix corresponding to `?X` is

```
<T-cell leukemia : is a : non-Hodgkin's lymphoma>^1.0
```

Similarly for the `?Y` antecedent:

```
<non-Hodgkin's lymphoma : is a : leukemia>^1.0
```

The instantiated `?Z` antecedent is an empty matrix associated with `leukemia`, though. The similarity vector S is $(0.8\bar{3}, 1, 1)$ and the aggregated $F(S)$ similarity equals $0.9\bar{4}$ than when using arithmetic mean as the OWA operator. Therefore the result of the rule consequent application is approximately

```
<T-cell leukemia : is a : non-Hodgkin's lymphoma>^1.0 AND
<T-cell le-ukemia : is a : leukemia>^0.989
```

when using the dynamic weighted mean in the $\Delta_{0.0\bar{5},1.0}$ update of the **A** concept by the rule consequent.

The query `<?X : isA : ?Y>`^{1} *AND* `<?Y : isA : leukemia>`^{1} on K would yield the instance vector $S = (0.8\bar{3}, 1)$ and $F(S) = 0.91\bar{6}$ when using arithmetic mean as F. The retrieved instances w.r.t. K would then be

`<T-cell leukemia : is a : non-Hodgkin's lymphoma>`$^{0.7\bar{3}}$ *AND*
`<T-cell le-ukemia : is a : leukemia>`$^{0.7\bar{3}}$

for `?X` and

Algorithm 2 The *applyRules*() function

Require: B, K, ϵ – rule set, concept set and threshold, respectively
Require: the *getInit*(), *applyRule*() functions
1: $applied \leftarrow \emptyset$
2: **for** $\mathcal{R} \in B$ **do**
3: $v, C \leftarrow getInit(\mathcal{R}, K)$
4: $V.init()$ {initialisation of the V structure}
5: **for** $var \in \mathcal{R}.variables$ **do**
6: $V[var] \leftarrow$ **nil**
7: **end for**
8: $applied \leftarrow applied \cup applyRule(\mathcal{R}, v, C, V, K, \epsilon)$
9: **end for**
10: **return** $\bigcirc(applied)$

`<non-Hodgkin's lymphoma : is a : leukemia>`$^{0.91\bar{6}}$

for `?Y`.

Multiple rules are evaluated using the *applyRules*() function, described in Algorithm 2. The *getInit*() function selects initial variable v and candidate set c to be passed to *applyRule*. v is an arbitrary antecedent subject variable such that all other antecedent variables are accessible from it via the subject–object references (for instance, `?X` can be initial variable for the rule `<?X : isA : ?Y>`^{1} AND `<?Y : isA : ?Z>`$^1 \rightarrow^1$ `<?X : isA : ?Z>`^{1}, while `?Y` and `?Z` cannot). C is a set of all concepts from K satisfying the v antecedent concept (i.e. having the same set of properties and referring to the same definite objects if the v antecedent concept actually has any).

Since the materialised rules introduce changes into the knowledge base, we must ensure that further possible materialisations based on these changes are computed as well. This is realised by a fixed-point *closure*() computation specified in Algorithm 3. The *stable*() function tests the termination of the algorithm by comparing the sets of concepts before and after the last rule set application. Based on the number and degree of changes among the concepts, it decides whether the update is stable or not already w.r.t. the ϵ_2 threshold. Various implementations of the *stable*() function are possible, resulting in different convergence properties and also degrees

of completeness of the eventual closure. For instance, it is possible to base the stability measure either on the σ concept similarity or on the difference of concept sizes[9] between consequent iterations. One can also combine these measures with relaxing the stability threshold proportionally to the number of iterations so far, thus ensuring fast convergence even for very large knowledge bases[10] (sacrificing completeness in general, though).

Algorithm 3 The *closure*() function

Require: $B, K, \epsilon_1, \epsilon_2$ – rule set, concept set and thresholds, respectively
Require: the *applyRules*(), *stable*() functions
1: $K_{tmp} \leftarrow K$
2: $C \leftarrow applyRules(B, K, \epsilon_1)$
3: **while NOT** $stable(C, K_{tmp}, \epsilon_2)$ **do**
4: $K_{tmp} \leftarrow \bigcirc(K_{tmp} \cup C)$
5: $C \leftarrow applyRules(B, K_{tmp}, \epsilon_1)$
6: **end while**
7: **return** $\{\mathbf{X} | \mathbf{X} \in \bigcirc(K_{tmp} \cup C) \wedge \mathbf{X} \notin K\}$ {the $\in, \notin$ (non)inclusion determined using $=$, i.e., the strong concept equality}

9.3.3 Notes on the Theoretical Principles' Implementation

Following Definition 1, it is quite natural to implement separate data structures – lexicon and knowledge base – catering for the ind_K, λ_K mappings and the K set, respectively. It allows for optimal space consumption and for clear modularisation of the following features:

– grounding of concepts in the natural language domain via the λ_k mapping;
– manipulation with concept matrices encoded using the unique indices only.

That means, however, that the knowledge addition and exploitation processes both have to be two-step: first, the lexicon structure has to be updated or consulted regarding the term to index mapping, and only then the knowledge base can be updated or queried.

The only case involving parallel usage of both structures in the EUREEKA implementation is resolution of polysemy. It occurs, e.g. when terms appear both as a predicate (a specific `property` type) and as a subject or object (a generic "top" `concept` type). The polysemy resolution is based on the types of concepts corresponding to terms for which the knowledge base has to be checked within the process of term to index mapping. For instance, the `lead` term in the `< group`

[9] Computed as $|\mathbf{A}| = |(i,j)|a_{i,j} \neq 0|$ for a concept $\mathbf{A}$.
[10] Very large means hundreds or thousands of concepts and millions of respective statements, or more.

`leaders : lead : groups >`[1] statement is to be mapped to an index bound to a concept having the `property` concept among its types. On the other hand, the same term's index regarding the `< lead : is a : metal >`[1] statement should be bound to a concept *not being* a sub-type of `property`.

For the knowledge base implementation, we use a relational database back-end for storing particular statements with their document-provenance information and other contextual meta-data (e.g. relevance scores). The conceptual matrices are dynamically constructed from the back-end within execution of the inference or querying algorithms. Such a solution is very efficient due to the state-of-the-art database indexing and query optimisation technologies. The statements stored in the database are associated with relevance scores. The scores are computed using an adapted version of the HITS algorithm [34]. They allow for anytime processing [35] of the EUREEKA data, thus further contributing to the framework's efficiency and scalability. Detailed description of the outlined knowledge base implementation is out of the scope of this chapter; however, it can be found in [36].

9.4 Processing the Publication Data

The previous section introduced the theoretical groundwork we have proposed in order to enable efficient processing of the emergent knowledge extracted from publications. Now we can move on to more practical issues, namely to the description of the delivered prototype for knowledge-based publication search. In the following we describe the particular data and method we employed when implementing CORAAL, our proof-of-concept prototype of knowledge-based publication search engine.

9.4.1 Data

Input As of March 2009, we have processed 11,761 Elsevier journal articles from the provided XML repositories that were more or less related to cancer research and treatment. The domain was selected due to the expertise of our sample users and testers from Masaryk Oncology Institute in Brno, Czech Republic. We processed articles evenly distributed across the journals in the following list[11]: 1. *FEBS Letters* (context: biochemistry); 2. *Biochemical Pharmacology* (context: pharmacology); 3. *Cancer Genetics and Cytogenetics* (context: oncology); 4. *Cell* (context: cell research); 5. *Trends in Cell Biology* (context: cell research); 6. *Experimental Cell Research* (context: cell research); 7. *Controlled Clinical Trials* (context: clinical medicine); 8. *Molecular Aspects of Medicine* (context: clinical medicine);

[11] Each of the journals was associated with a specific context identifier to maintain the sub-domain provenance of the respective extracted information and reflect it later on in the CORAAL user interface.

9. *Advanced Drug Delivery Reviews* (context: pharmacology); 10. *Gene* (context: genetics); 11. *Trends in Genetics* (context: genetics); 12. *Genomics* (context: genetics); 13. *Leukemia Research* (context: oncology); 14. *Journal of Microbiological Methods* (context: biology); 15. *Trends in Microbiology* (context: biology); 16. *Journal of Molecular Biology* (context: biology); 17. *Oral Oncology* (context: oncology); 18. *European Journal of Pharmacology* (context: pharmacology). From the article repository, we extracted the knowledge and publication meta-data for further processing by CORAAL. Besides the publications themselves, we employed legacy machine-readable vocabularies[12] for refinement and extension of the automatically extracted knowledge.

Output CORAAL exposes two data sets as an output of the publication processing:

- First, we used a *triple store* containing publication meta-data (citations, their contexts, structural annotations, titles, authors and affiliations) associated with respective full-text indices. The resulting store contained 7,608,532 of RDF subject–predicate–object statements [37] describing the input articles. This included 247,392 publication titles and 374,553 authors (both from full-texts and references processed).
- Second, we employed a custom EUREEKA **knowledge base** with facts of various certainty extracted and inferred from the article texts and the seed life science thesauri. Directly from the articles, 215,645 concepts were extracted (and analogically extended). Together with the data from the initial thesauri, the domain lexicon contained 622,611 terms, referring to 347,613 unique concepts. The size of the emergent knowledge base was 4,715,992 weighed statements (ca. 99 and 334 extracted and inferred statements per publication in average, respectively). This number is significantly smaller than in the case of the semifinal prototype. However, this is due to a full integration of the knowledge from formerly separate contexts, the data themselves are still the same. The contextual meta-knowledge related to the statements (like provenance information) amounts to more than 10,000,000 additional statements should it be expressed in RDF triples.

Thanks to the improved knowledge representation back-end, generation of the output data sets from the input articles took 2 days (as opposed to 4 days in the semifinal CORAAL prototype). Query evaluation on the produced content takes usually fractions and at most units of seconds.[13]

[12] The NCI and EMTREE thesauri – see http://www.cancer.gov/cancertopics/terminologyresources and http://www.embase.com/emtree/, respectively.

[13] These results were achieved on a single server machine (which is not exclusively dedicated to CORAAL). There are still reserves regarding scalability even with the current implementation; however, for processing data two and more orders of magnitude larger, a distributed solution would be much better.

9.4.2 Method

The CORAAL tool shares certain features with its namesake, the coral organism. It *feeds* on publication text and meta-data, *digesting* it and *merging* it together with the content consumed so far. In this continuous process, it builds up an emergent basis of the processed knowledge. The knowledge is then being *exposed* as a support for the ecosystem of particular research communities. The following sections describe the particular phases of the CORAAL digestive process.

Munching on the Publications CORAAL processes the articles in the form of respective XML files provided by Elsevier, B.V., within the Elsevier Grand Challenge.[14] The output of XML parsing is twofold:

– raw text files as a basis for the emergent knowledge extraction
– RDF [37] data conforming to the SALT [38] ontology schemata,[15] capturing

- document meta-data, i.e. authors, titles, references (the SALT annotation ontology)
- document text blocks, i.e. paragraphs (the SALT document ontology)
- document rhetorical blocks, i.e. abstracts in the current implementation (the SALT rhetorical ontology)

The extracted RDF proceeds directly to the next phase (Section 4.2); however, the raw texts undergo further processing in order to extract initial snippets of knowledge from them. For the current proof-of-concept CORAAL prototype, we use a simple, yet already quite productive NLP-based heuristics for the extraction of knowledge in the form of subject–predicate–object triples[16]:

1. we identify sentences in the raw text and split them in to particular word tokens
2. we tag the words in sentences by a probabilistic part-of-speech tagger
3. we chunk-parse the tagged sentences using a generic probabilistic shallow parser; the resulting plain (i.e. non-nested) chunks are of three types: *NP* (noun phrase – noun sequences, possibly with modifiers and/or grouped by coordinate conjunctions), *VP* (verb phrase – verbs only), *PP* (prepositional phrase – prepositions only)

[14] See http://www.elseviergrandchallenge.com/.

[15] See http://salt.semanticauthoring.org/onto/. An extracted RDF file example is given at http://resources.smile.deri.ie/coraal/2008/11/ee7c3ec2536e6754ad424c9f95a0d8dce7059a4e.rdf.

[16] The heuristics is quite similar to the technique described in [39]. We use the Python NLTK library for NLP (see http://nltk.sourceforge.net). We also experimented with state-of-the-art ontology learning solutions (such as Text2Onto, see http://ontoware.org/projects/text2onto/). The respective tools performed rather poorly in larger scale, though, while providing not that significant improvement in quality when compared to our simple approach. However, we do plan to include more sophisticated as well as domain-specific methods of knowledge extraction (cf. [4, 7, 40]) into our light-weight implementation at some stage.

4. for every *NP* (*PP NP*)* *VP PP*? *NP* (*PP NP*)* chunk sequence[17] present in a sentence, we assume that the part preceding the verb phrase expresses a subject, the succeeding part an object and the verb phrase itself a predicate (i.e. property or relation holding between the subject and the object)
5. we use several additional heuristics in order to generate the actual triple terms:
   - the head verb of *VP* is used as the predicate term (a preferred name based on WordNet [41] verb synsets used if possible)
   - if the predicate part of the chunk sequence is in the *VP PP* form, the *VP* head verb with the respective preposition is used as the predicate term; if *VP* without a consequent *PP* is followed by *NP* (*PP NP*)$^+$, the head noun of the first *NP* and *PP* preposition is attached to the predicate terms in order to form a more specific predicate; an additional triple expressing the *is a* relation between the specific and verb-only generic predicate is generated
   - the remaining *NP* (*PP NP*)$^+$ sequences are merged together to provide a basis for the subject and object term construction
   - if there are modifiers or other nouns attached to a head noun, additional triple is generated in order to capture the *is a* relationship between the modified noun and the noun itself (based on heuristics discussed for instance in [42])
   - if there is an enumeration of terms in a noun phrase, additional triples are generated in order to capture mutual negative *is a* (*not a*) relationships (i.e. disjointness; based on a heuristics explained for instance in [43]); enumerations in either subject or object noun phrase result in multiple basic triples, too

Example 9 To give an example, the sentence:

> Mutated genes and cancer proteins play role in the neoplasm development.

would yield the basic triples:

```
(cancer proteins, play role in, neoplasm development), (mutated
genes, play role in, neoplasm development)
```

and the additional triples:

```
(mutated genes, is a, genes), (cancer proteins, is a, proteins),
(mutated genes, not a, cancer proteins), (cancer proteins, not a,
mutated genes), (neoplasm development, is a, development), (play
role in, is a, play)
```

To each of the extracted triples, we attach a relevance score in the [0,1] interval. The score is computed as $\nu_1 f_P(t_\mathrm{p}) \nu_2 \frac{1}{2}(f(t_\mathrm{s}) f_D(t_\mathrm{s}) + f(t_\mathrm{o}) f_D(t_\mathrm{o}))$, where $f(t)$ is

[17] *, $^+$ and ? mean zero or more, one or more and zero or one repetitions of the preceding expression, respectively.

an average absolute frequency of the term t per publication, $f_D(t)$ is the number of publications containing the term t and $f_P(t)$ is the number of triples containing the predicate term.[18] t_s, t_p and t_o stand for the subject, predicate and object terms, respectively. ν_1, ν_2 are normalising constants for the respective operands. The resulting `(subject, predicate, object, score)` quads are passed further to be digested.

Digesting the Content Digestion of the publication RDF extracted in the previous step is pretty straightforward. The particular models are assigned a URI derived from the SHA-1 sum.[19] of their titles and then they are stored as named graphs in an RDF repository.[20] In parallel to the process of graph insertion into the repository, indices of authors, co-authors and publication references are updated. The publication content, titles and author names are being indexed using Lucene.[21] to facilitate efficient full-text searching among the respective data and meta-data. Details of the whole process are described in [21, 22].

The initial knowledge extracted from the text is processed using our EUREEKA library – a proof-of-concept implementation of the principles introduced in Section 9.3 – after transforming the quads into concept matrices. Note that the mapping of the lexical terms to the concept indices is tackled by the lexicon structure specified in Section 9.3.3, which also takes care of features like synonymy or polysemy resolution.

Example 10 Considering the `mutated genes` subject from Example 9, the quads corresponding to the respective concept would be

```
(mutated genes, is a, genes, 1), (mutated genes, not a, cancer pro-
teins, 1), (mutated genes, play role in, neoplasm development, 1)
```

After import into EUREEKA, we would get the following matrix[22]:

mutated genes	mutated genes	genes	cancer proteins	neoplasm development
is a	0.0	1.0	-1.0	0.0
play role in	0.0	0.0	0.0	1.0

Similarly for the other subjects (`cancer proteins`, `neoplasm development`, etc.).

[18] We exclude the by far most common `is a` predicate from the set considered for the $f_P(t)$ computation (the value of $\nu_1 f_P(t_p)$ was set to 0.8 for the `is a` statements). We also do not include the statements with $f_P(t_p) = 1$ at all. Note that $f(t_s)f_D(t_s)$, $f(t_o)f_D(t_o)$ are relevance scores of the particular `s`, `o` terms, respectively.

[19] See http://en.wikipedia.org/wiki/SHA_hash_functions.

[20] We use the Sesame repository (see http://www.openrdf.org/).

[21] A comprehensive Java search engine library (see http://lucene.apache.org).

[22] Modulo mapping the terms to indices and neglecting the infinite number of columns and rows with zero-only elements. We consider `not a` as a negation of `is a`.

For the actual emergent knowledge processing in CORAAL, we use the following basic "ontology": `<p : isA : Property>`[1], where $p \in \{TransProp, ASymmProp, SymmProp\}$ for the transitive and (anti)symmetric property, respectively; `<isA : isA : TransProp>`[1], `<isA : isA : ASymmProp>`[1]; `<partOf : isA : TransProp>`[1]; `<sameAs : isA : TransProp>`[1], `<sameAs : isA : SymmProp>`[1]. We also have to model the learned disjointness, which is done similarly to Example 10, i.e. by specifying `<x : isA : y> -1` and `<y : isA : x> -1` for the disjoint entities x, y.

The simple basic ontology merged with the EMTREE and NCI life science thesauri[23] serves as a seed model for the incorporation of the statements extracted from publications. The semantics of the seed model was furthermore extended by RDFS entailment rules [26] ported to the format specified in Definition 6 (assuming crisp 1 degrees for the respective rule statements).

The emergent knowledge digestion itself makes use of a so-called *ACE* pipeline[24] based on our EUREEKA library:

- *A* for *addition* – the extracted quads are incrementally added to the initial knowledge base (as one concept per one quad)
- *C* for *closure* – after the addition of new facts into the knowledge base, we compute its closure using a fixed-point evaluation of a given rule set
- *E* for *extension* – the extracted concepts are analogically extended by new properties from similar concepts present in the knowledge base after the closure

$\delta_H(\mathbf{A}, \mathbf{B}) = \frac{1}{|H|} \sum_{(i,j) \in H} |a_{i,j} - b_{i,j}|$ distances under the open world assumption are used as a basis for all the similarity computations in the pipeline. Arithmetic mean and weighted arithmetic mean are used in the $\bigcirc$, $\Delta_{u,v}$ operators, respectively. The $\Delta_{u,v}$ operator *u, v* parameters are set to 1, 0.8 in the *addition* step.

The content of the knowledge bases produced by the ACE pipeline is linked with the publication URIs in the document RDF repository (using an index tracking provenance of particular extracted subjects and objects). Conversely, publications are linked to respective significant concepts in the knowledge bases, therefore one may conveniently get from the knowledge to the relevant publications and vice versa.

---

[23] See http://www.embase.com/emtree/ and http://www.cancer.gov/cancertopics/terminology resources, respectively. EMTREE terms and relations were used in case of conflicts, since they cover more general domain. Synonyms defined in the thesauri were reflected in the lexicon data structure accordingly.

[24] Note that the pipeline can be executed even as $(ACE)^+$, i.e. as a search for a global fixed point of the respective operations; however, for the CORAAL prototype we employed only single iteration, since the results were already sufficient for the presented proof-of-concept.

9.5 Using CORAAL

For the user interface of our system, we employed the MIT's state-of-the-art Exhibit framework (cf. http://www.simile-widgets.org/exhibit/). It supports faceted browsing of the knowledge-based search results, letting users to conveniently focus on the relevant answers. Similarly, we allow for faceted browsing of the classical full-text search results that are tightly integrated with the knowledge extracted or inferred from the respective articles.

Examples of particular queries possible in CORAAL for various types of search are as follows:

– publication **knowledge** search:

- *concepts* – use just the concept name, i.e. respective term(s). Examples: `lymphoblastic leukemia`, `chemosensitizer`
- *statements* – use the `S : P : O` syntax, where `S, P, O` are subject, predicate, object expressions, respectively. The expressions may be either in the form of a concept name or in the form of a variable (anything starting with `?`, possibly even `?` alone. The only limitation is that there may be at most one variable in a query statement. Check for concepts satisfying a feature or for relations between concepts. Example: `? : is a : breast cancer, p53 : ? : early carcinogenic events, rapid antigen testing : part of : ?`
- *conjunctive statements* – use the $\texttt{St}_1$ `AND` $\texttt{St}_2$ `AND ... AND` $\texttt{St}_n$ syntax, where $\texttt{St}_1, \ldots, \texttt{St}_n$ are statements. At most one variable identifier is allowed to appear in a conjunctive query. Check for concepts satisfying multiple features. Examples: `rapid antigen testing : part of : ? AND ? : is a : clinical study, ? : blocks : binding site AND ? : mediator of activation : reactive oxygen metabolite`
- *negative statements* – use the `NOT St` syntax, where `St` is a statement. The `NOT` keyword may be used even inside the statement, e.g. before the predicate name. Check for concepts explicitly satisfying a negative feature. Examples: `NOT ? : is a: penicillin, acute granulocytic leukemia : NOT is a : chronic neutrophilic leukemia`
- *complex queries* combining the above. Examples: `? : NOT is a :mouse AND ? : is a : animal, ? : as : complementary method AND ? : NOT type : polymerase chain reaction`

– publication text, title and author **full-text** search:

- terms in the Lucene full-text search syntax, i.e. term names plus wild-cards like * or ? and boolean keywords like `AND, OR` or `NOT`.[25] Examples: `"breast cancer", carci*, "breast cancer" AND p53`

[25] Detailed Lucene syntax description can be found at http://lucene.apache.org/java/2_3_2/queryparsersyntax.html. Note that even though the meaning of the `AND, NOT` keywords is intuitively

CORAAL itself can be accessed at http://coraal.deri.ie:8080/coraal/.[26] The following browsers have been tested with CORAAL and are known to work on most desktop configurations and operating systems:

– Firefox – versions 2.x, 3.x and newer
– Internet Explorer – versions 7.x and newer (in most cases only on Windows Vista, though)
– Opera – versions 9.6 and newer
– Safari – versions 3.1 and newer
– Google Chrome – all versions

After pointing your browser to the URL, the main search interface will appear as shown in Fig. 9.3. The tabs correspond to the particular types of search – the *Knowledge* tab serves for publication knowledge search using the query language specified above, while the *Text, Title, Authors* tabs realise full-text search for the respective publication (meta)data.

Figure 9.3 shows how a query for *knowledge* is constructed simply by typing it into the search box. You can also use an interface for assisted query construction with auto-completion capability, as seen in Fig. 9.4.

The answers are displayed as particular statements provided with several types of meta-information:

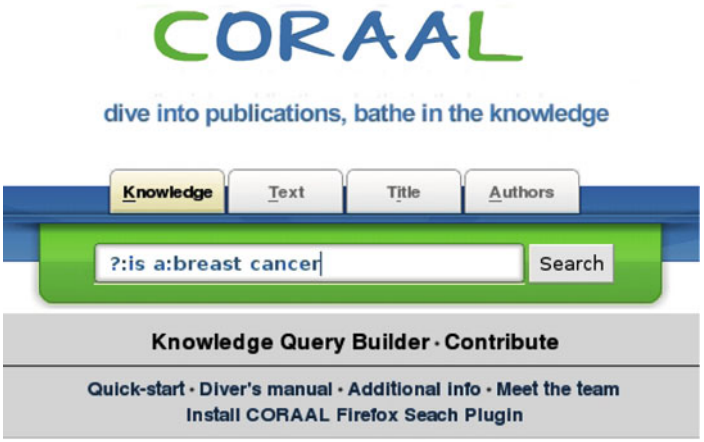


Fig. 9.3 Asking a query – direct

similar for both types of search in CORAAL, the knowledge and full-text variants are based on completely different principles. For instance, `NOT` indicates documents not containing the query expression for the *full-text* search, while in the *knowledge* search, it leads to documents containing a negation of the respective query statement; similarly for the `AND` keyword.

[26] Note that you can watch a video comprehensively illustrating the essential CORAAL capabilities at http://resources.smile.deri.ie/coraal/videos/coraal_web.mp4 before starting to play with the tool itself.

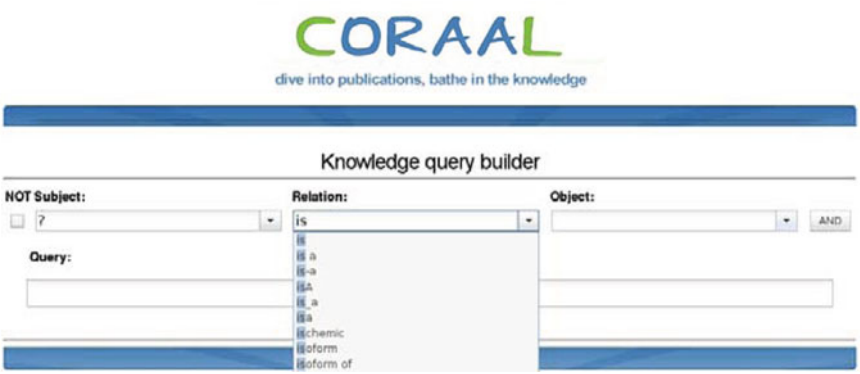


Fig. 9.4 Asking a query – assisted

– *source* provenance – articles relevant to the statement
– *context* provenance – sub-domain of life sciences the statement relates to (determined according to the main topic of the journal that contained the articles the statement was extracted from)
– *certainty* – how certain the system is that the statement holds and is relevant to the query (values between 0 and 1)
– *inferred* – whether the statement was inferred or not (i.e. directly extracted)

You can filter the answer statements based on their particular elements (subjects, properties and objects), associated meta-information and their negativity. This is done using the respective filtering boxes as shown in Fig. 9.5. Using a particular

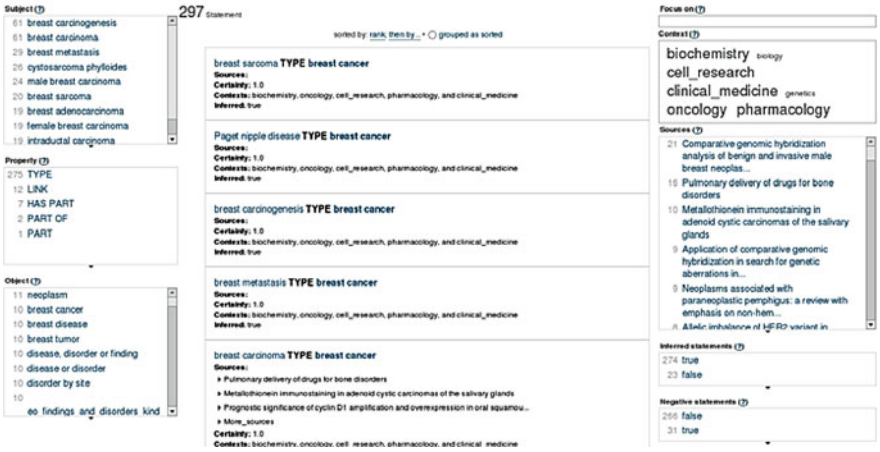


Fig. 9.5 Answer display – full

filtering, you can quickly focus only on statements of your interest. Such a specific focus can be seen in Fig. 9.6[27] Article provenance summaries of particular statements can be displayed in-line as shown in Fig. 9.7.

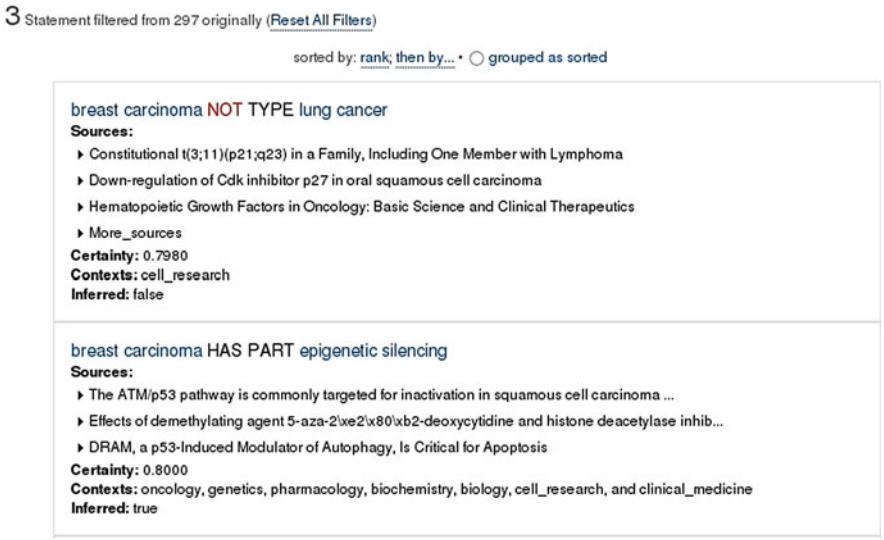


Fig. 9.6 Answer display – focused

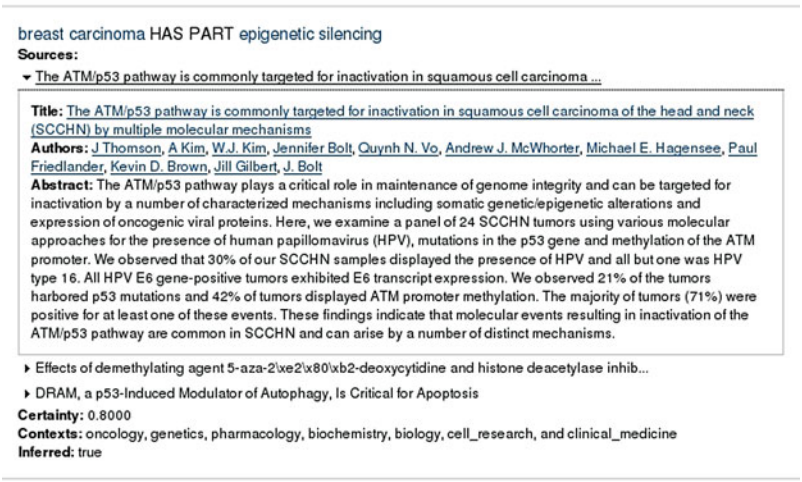


Fig. 9.7 Answer display – in-line provenance info summary

[27]Note that the `HAS PART` relation has rather general semantics in the knowledge extracted by CORAAL, i.e. its meaning is not strictly mereological in the physical sense, it can refer also to, e.g. conceptual parts or possession of entities. Similarly for the `PART OF` relation.

Classical full-text search results in paragraphs containing the queried terms. Links to the respective publication details are provided, as well as citation contexts for the queried terms (i.e. publications referenced from the adjacent text). Fig. 9.8 shows the publication *text* search results. Several filtering boxes are present again, such as authors or citation contexts. One can filter publications according to the concepts they contain, too, but also according to the indirect (inferred) general topics or specific instances associated with the articles within the CORAAL knowledge base.

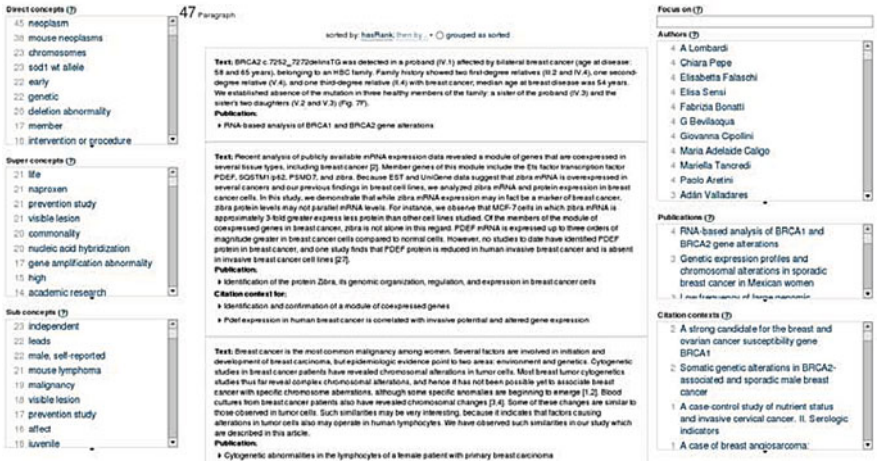


Fig. 9.8. Classical full-text search results

Clicking on an article link displays the respective publication details view (Fig. 9.9). One can browse associated meta-data (e.g. authors, abstract or references) there. List of related publications computed based on the overlapping knowledge is available as well. Using the *concepts* tab, it is possible to browse the knowledge associated with the particular article as per directly or indirectly related concepts.

The author search can be filtered, too. Figure 9.10 shows list of authors corresponding to the "Lin" name filtered only to those who have written an article concerned with the "gene amplification abnormality" topic. This feature of CORAAL can be used, for instance, for finding candidate experts on certain topics.

9.6 Preliminary Tests with Domain Experts

Before the final stage of the current CORAAL prototype development, we arranged four sample users with different backgrounds:

– a computer scientist with limited knowledge of medical terminology
– a computational linguist with strong experience in bioinformatics projects
– a person with strong background in both computer science and biomedicine
– a medical oncologist with no background in computer science or IT R&D

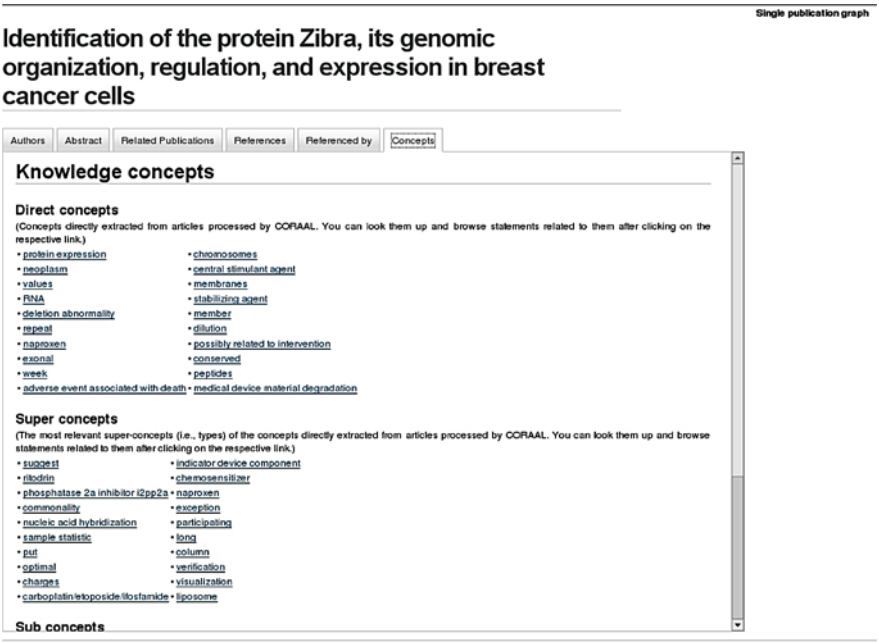


Fig. 9.9. Publication details

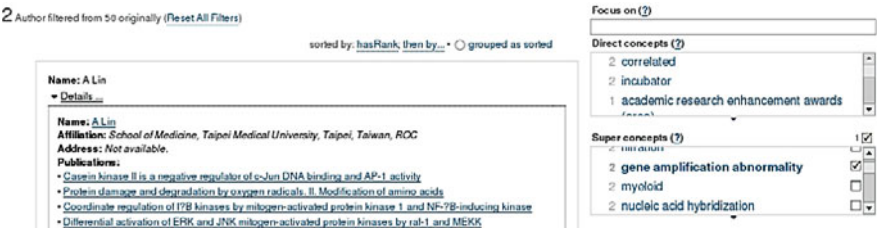


Fig. 10.10. Filtering author search

We prepared five tasks to be worked out with both CORAAL and a base-line application (ScienceDirect or PubMed). Our hypothesis was that the users should perform better with CORAAL than with the base-line, since the tasks were focused rather on structured knowledge than that on a plain text-based search.[28]

The average level of evaluation tasks' direct similarity to the day-to-day agenda of users was approximately 4 on the 1–6 scale (from least to most relevant), meaning that the tasks had tangible relation to the practice. The success rate of task

[28]For instance, the users were asked to find all authors who support the fact that the `acute granulocytic leukemia` and `T-cell leukemia` concepts are disjoint, or to find which process is used as a `complementary method`, while being different from the `polymerase chain reaction`, and identify publications that support their findings.

accomplishment was 60.7 and 10.7% when using CORAAL and the base-line application, respectively. This clearly confirms our hypothesis.

However, the users were quite negative in the assessment of CORAAL usability in close relation to the tasks they were not able to tackle. After a detailed analysis and additional interviews, we found out 53.9% of failures were due to a user's critical mistake, while 46.1% were due to particular CORAAL deficiencies (which were all remedied to large extent as of now). For users who requested additional education instead of mere relying on the provided online materials, the mistakes were reduced by about 75%, the performance was much better and the frustration diminished.

Particular feedback included not only the following:

> Potentially very useful for searching facts across documents...
> ...useful to get a quick overview of the connections between concepts...
> I got the information I wanted for "What is an aneurysm?"...
> ...could be very useful for education of medical students and starting researchers...

but also the following:

> results are often too general or not relevant...
> when I open this result set it is empty...
> For me it is obvious that epithelioid is not a type of sarcomatoid...
> ...I have no clue what can be used as predicates in relations...

We did a detailed analysis of the user feedback and organised two workshops with more oncology researchers and practitioners involved to help us improve the system. Two most critical issues were identified:

1. the scattered and loosely integrated presentation of the knowledge search results (the former CORAAL interface was displaying the results per a context, not in one list of resulting statements);
2. lack of guided knowledge query construction that would take the actual knowledge base content into account.

A remedy for these issues was implicitly or explicitly demanded by all the sample users participating in the CORAAL evaluation or (re)development. The former has been addressed by the currently implemented back-end and consecutive integral display of the results. The latter has been resolved within the knowledge query builder form with context-sensitive auto-completion. Both solutions were unequivocally considered by the sample users as fully implementing their requests. Moreover, due to the improvements and increased intuitiveness of the interface, the users were able to perform up to six times faster and 40% more efficiently than with the old CORAAL version. They were also able to use the tool after a 2-min presentation of the query language, relying only on the online contextual help in the user interface from then on.

Less critical, but still important were requests for extensions of the query language (mainly regarding previously disallowed variables in the predicate position

enabling direct exploration of arbitrary relations between particular concepts). As shown in Section 9.5, the current syntax has been extended accordingly.

The expert users also had slight problems with too general, obvious or irrelevant results presented. These concerns were addressed by the following particular improvements:

- improved relevance-based sorting of concepts and statements – more relevant statements present in the top results;
- the intuitive faceted browsing and filtering functionality of the current user interface – support for fast and easy reduction of the displayed results to a subset with certain features (i.e. statements having only certain objects or authors writing about certain topics).

The improvements we made were considered as mostly sufficient regarding the sample users' concerns (an average $4.\overline{6}$ score on the 1–6 scale going from least to most sufficient).

9.7 Related Work

Here we elaborate on particular approaches related to the core scientific contribution of our work (i.e. EUREEKA, the emergent knowledge processing framework). Conformance of EUREEKA and CORAAL regarding the Semantic Web standards is discussed then, too.

9.7.1 Similar Approaches to Emergent Knowledge Processing

Bechhofer et al. [1] examine *combination of automatic knowledge extraction with ontology engineering and debugging* in order to address the knowledge acquisition bottleneck; however, the approach does not provide any automatic means for how to deal with the noisy and shallow knowledge – significant human efforts still have to be spent before it can be properly and reliably exploited by the traditional tools. *Combination of ontology learning and reasoning* was described in [8], but the work deals much rather with the resolution of inconsistencies in learned knowledge and consequent essentially trivial classical inference; no full-fledged reasoning tailored to the learned knowledge is proposed. Maedche [5] looks into perspectives of *emergent semantics for learned ontologies*, but it merely bridges the human-oriented and machine-oriented layers of the knowledge representation and emphasises the important bottom-up role of ontology learning in the process of the emergent semantics construction. Ottens et al. [6] research *clustering-based emergent construction of ontologies* from text managed by a multi-agent self-organising system. However, no mechanisms for robust and meaningful integration of expressive learned ontologies beyond set union or statistical concept clustering are offered either in [5] or in [6]. Moreover, no formal notion of semantics that

would allow for complex universal inference and/or automated refinement of the ontologies is elaborated by these approaches.

In terms of related *knowledge representation principles*, conceptual spaces [44, 45] and optimality theory [46] both complement logics-based approaches and aim at bridging sub-conceptual machine-learning (connectionist) and symbolic paradigms in formal knowledge representation, being inspired by the layered structure of the human cognition. However, the connectionist modules assumed as inputs for the meso-level representations being proposed in these approaches are far from being universally applicable across possible domains of emergent knowledge processing. Rather intricate principles of the proposed formalisms and more or less explicit reliance on the logical approaches at the symbolic level also complicate easy user involvement. In all these respects, our approach is more applicable for the processing of arbitrary emergent knowledge.

Regarding related *memory-based approaches* to reasoning, [32] concerns queries evaluated by returning similar records from an in-memory database on a parallel architecture, exponentially reducing the search time. Furthermore, we partly share motivations with the memory-based *analogical framework* presented in [33], which develops an agent-based architecture for connectionist construction of analogies among sets of episodes and hypotheses. However, none of the approaches allows for combination of automatically extracted data with intuitive user involvement in the definition of the actual semantics of the data. This makes our approach more flexible and applicable in many practical domains where the knowledge acquisition bottleneck is a crucial issue.

Sowa's conception of *analogical reasoning* operating on the top of possibly *automatically extracted conceptual graphs* [47] is very close to our approach. However, it implicitly assumes transformation of the emergent knowledge into precise conceptual graphs, which reduces the range of possible inputs a lot. Quite obviously, we cannot expect that the automatic NLP or KDD methods will always produce meaningful and expressive enough results and [47] does not contain any hints on what to do with such inputs in order not to pollute the consequent inferences. Moreover, conceptual graphs are equivalent to (a subset of) crisp predicate logics and even the rather ad hoc strategies for coping with dynamic, vague or inconsistent data proposed in [48] do not ensure as robust and comprehensive straightforward deployment as our approach.

9.7.2 Conformance to the Semantic Web Standards

The KONNEX part of CORAAL is directly conforming with regard to the RDF [37] standard, since all the extracted publication data and meta-data is stored, processed and exposed as quads via an RDF store. The publication knowledge is processed using our EUREEKA framework, which builds on a custom alternative knowledge representation tailored to noisy and uncertain emergent facts.

However, as can be seen from Definitions 1 and 6 in Section 9.3, the knowledge representation format we use is compatible with RDF triples and rules encoded

using RDF triples (e.g. RDFS entailment rules [26]), which can thus be straightforwardly imported (while setting respective degree values to 1). Due to the support for negative statements in EUREEKA, we are also able to represent an incomplete set of the OWL [49] features, namely those covered by the pD* formalism [50] and/or OWL 2 RL profile [51].

The statements produced as a result of empirical knowledge base querying or look-up in CORAAL are compatible with RDF triples up to the degrees. These can, however, be captured in RDF as well according to the W3C working group note [52]. The particular interpretation of the degrees can then be specified using the uncertainty annotation ontology delivered by the W3C Uncertainty Reasoning incubator group as a part of [53].

9.8 Conclusion and Future Work

We have delivered a solid and self-contained piece of innovative work in the form of the CORAAL prototype and the technologies that power it. The two major challenges we targeted were as follows:

1. *scientific* challenge – practical, scalable and cost-effective acquisition of machine-readable knowledge. The light-weight empirical knowledge representation framework introduced in Section 9.3 caters for the emergent knowledge extracted from publications in a simple, yet already quite useful way.
2. *engineering* challenge – integration of the classical and semantic publication search. The CORAAL prototype described in Sections 9.4 and 9.5 coherently integrates KONNEX – an implementation of a semantic publication repository with full-text and meta-data search functions – and EUREEKA – an implementation of the knowledge-based search. CORAAL also comes with a comprehensive and seamlessly integrated web interface conveniently exposing the KONNEX and EUREEKA functionalities to the users.

We have processed a non-trivial amount of data purely automatically with CORAAL and the indicative tests with real users have proven that we are on the right path regarding our vision. Already now we are able to effortlessly extract and process the knowledge hidden in large legacy repositories and offer it conveniently to the users who seek for it.

Regarding a particular CORAAL application, we have identified the following possibilities following a discussion with our sample users:

– *knowledge-based retrieval* of publications – For example, finding articles describing an arbitrary relation between a cancer type and a gene. This is apparently the most obvious and straightforward application of CORAAL.
– *automated tagging* of articles – Supported by the association of the most relevant topics (i.e. super-type concepts) to publications. Note that the tags come both

from the seed thesauri and from the most general concepts in the article corpus. Basically any life science vocabulary can be incorporated for the tagging together with or instead of the currently used NCI and EMTREE thesauri.
- rudimentary automated *expert finding* – Filtering of authors based on the topics they write about.
- semi-automated *population of the standard biomedical vocabularies* by the publication knowledge – Directly supported by the integration of the extracted statements into the seed domain thesauri. Further manual curation would be needed for the exported content, though, to tackle possible noise.
- application of CORAAL as a *general-purpose publication knowledge back-end* with arbitrary services implemented on the top of it – For example, textual entailment service checking for whether the statements present in article A are consistent with those of article B and to which extent, or services extending state-of-the-art tools via annotations of their content by the statements extracted and processed by CORAAL.

The range of possible applications is wider for CORAAL than for any similar system we know of. The CORAAL applicability is extended by its high portability due to the automation of the knowledge extraction, integration and refinement processes. This supports the prototype's indubitable potential for a further development into a truly production system.

However, two important things still remain to be accomplished:

1. *utilising the wisdom of the crowds* – support for intuitive and unobtrusive dynamic user involvement in the knowledge base updates, namely by (in)validation of existing statements, introduction of new statements and submission of new rules refining the emergent domain semantics
2. *making the step from CORAAL to a CORAAL reef* – proposal and implementation of a distributed peer-to-peer model covering multiple CORAAL installations autonomously communicating with each other (e.g. asking for answers when no answer is available locally or exchanging appropriate rules to improve the local semantics)

After incorporating the capabilities of the prospective CORAAL reefs into the ecosystem of the current online publishing, we can instantly help realise an exciting future of the semantic e-Science, building on the huge body of legacy knowledge possessed by the research institutions and scientific publishing houses.

Acknowledgments This work has been supported by the EU IST 6th framework's project "Nepomuk" (FP6-027705) and the "Líon" and "Líon II" projects funded by Science Foundation Ireland under Grant No. SFI/02/CE1/I131, SFI/08/CE/ I1380, respectively. We would like to thank the employees of Masaryk Oncology Institute for their feedback and to Ioana Hulpus for her work on the former CORAAL user interface. Very special thanks goes to the people who have actively participated in the continuous prototype evaluation and testing, namely to (in alphabetical order) Doug Foxvog, Peter Gréll, MD, Miloš Holánek, MD, Matthias Samwald, Holger Stenzhorn and Jiří Vyskočil, MD. We also acknowledge the valuable comments from the anonymous reviewers who helped to improve the final shape of the chapter.

References

1. Bechhofer, S., Gangemi, A., Guarino, N., van Harmelen, F., Horrocks, I. Klein, M., Masolo, C., Oberle, D., Staab, S., Stuckenschmidt, H., Volz, R.: Tackling the ontology acquisition bottleneck: An experiment in ontology re-engineering (2003) Retrieved at http://tinyurl.com/96w7ms, Apr'08. 13 Jul 2010
2. Gomez-Perez, A., Fernandez-Lopez, M., Corcho, O.: Ontological Engineering. Advanced Information and Knowledge Processing. Springer, New York (2004)
3. Aberer, K., Cudré-Mauroux, P., Ouksel, A.M.: Emergent semantics principles and issues. In: Proceedings of Database Systems for Advanced Applications, 9th International Conference, DASFAA 2004, Jeju Island, Korea (2004)
4. Maedche, A., Staab, S.: Ontology learning. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies. Springer, New York (2004) 173–190
5. Maedche, A.: Emergent semantics for ontologies. In: Emergent Semantics. IEEE Intelligent Systems. IEEE Press, NYC, USA (2002) 85–86
6. Ottens, K., Aussenac-Gilles, N., Gleizes, M.P., Camps, V.: Dynamic ontology coevolution from texts: Principles and case study. In: Proceedings of ESOE 2007 Workshop, CEUR-WS, Busan, Korea (2007) 70–83
7. Buitelaar, P., Cimiano, P.: Ontology Learning and Population: Bridging the Gap Between Text and Knowledge. IOS Press, Amsterdam, Netherlands (2008)
8. Haase, P., Völker, J.: Ontology learning and reasoning – dealing with uncertainty and inconsistency. In: Proceedings of the URSW2005 Workshop. (NOV 2005), Galway, Ireland 45–55
9. Hein, J., Hendler, J.: Dynamic ontologies on the web. In: Proceedings of AAAI 2000, AAAI Press, Menlo Park, California, USA (2000)
10. Haase, P., van Harmelen, F., Huang, Z., Stuckenschmidt, H., Sure, Y.: A framework for handling inconsistency in changing ontologies. In: Proceedings of ISWC'05. Volume 3792 of LNCS. Springer, New York (2005) 353–367
11. Straccia, U.: A fuzzy description logic for the semantic web. In: Sanchez, E. (ed.) Fuzzy Logic and the Semantic Web. Capturing Intelligence. Elsevier, Amsterdam (2006) 73–90
12. Flouris, G., Huang, Z., Pan, J.Z., Plexousakis, D., Wache, H.: Inconsistencies, negations and changes in ontologies. In: Proceedings of AAAI 2006, AAAI Press, Menlo Park, California, USA (2006)
13. Sheth, A., Ramakrishnan, C., Thomas, C.: Semantics for the semantic web: The implicit, the formal and the powerful. International Journal on SemanticWeb & Information Systems 1(1) (2005) 1–18
14. Frith, C.: Making Up the Mind: How the Brain Creates Our Mental World. Blackwell, Oxford, UK (2007)
15. Gentner, D., Holyoak, K.J., Kokinov, B.K. (eds.): The Analogical Mind: Perspectives from Cognitive Science. MIT Press, Cambridge, MA (2001)
16. McGuinness, D.L.: Ontology-enhanced search for primary care medical literature. In: Proceedings of the Medical Concept Representation and Natural Language Processing Conference, Phoenix, Arizona, USA (1999) 16–19
17. Abasolo, J.M., Gómez, M.: M.: Melisa: An ontology-based agent for information retrieval in medicine. In: Proceedings of the First International Workshop on the Semantic Web (SemWeb2000), Lisbon, Portugal (2000) 73–82
18. Dietze, H., et al.: Gopubmed: Exploring pubmed with ontological background knowledge. In: Ontologies and Text Mining for Life Sciences, IBFI (2008)
19. 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, USA (2003)
20. Müller, H.M., Kenny, E.E., Sternberg, P.W.: Textpresso: An ontology-based information retrieval and extraction system for biological literature. PLoS Biology 2(11) (2004) 1984–1998

21. Groza, T., Handschuh, S., Moeller, K., Decker, S.: KonneXSALT: First steps towards a semantic claim federation infrastructure. In: The Semantic Web: Research and Applications (Proceedings of ESWC 2008), Springer, New York (2008) 80–94
22. Hulpus, I.: Design and implementation of a semantic claim federation infrastructure. Master's Thesis, Technical University of Cluj-Napoca (2008)
23. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American 5 (2001)
24. Zadeh, L.A.: Fuzzy sets. Journal of Information and Control 8 (1965) 338–353
25. Ogden, C.K., Richards, I.A.: The Meaning of Meaning. Mariner Books (1989)
26. Brickley, D., Guha, R.V.: RDF Vocabulary Description Language 1.0: RDF Schema. (2004) Available at (Feb 2006): http://www.w3.org/TR/rdf-schema/. 13 Jul 2010
27. Deschrijver, G., Cornelis, C., Kerre, E.E.: On the representation of intuitionistic fuzzy t-norms and t-conorms. In: Transactions on Fuzzy Systems. IEEE (2004)
28. Yager, R.R.: On ordered weighted averaging aggregation operators in multi-criteria decision making. IEEE Transactions on Systems, Man and Cybernetics 18 (1988) 183–190
29. Greenwald, A.G.: Cognitive learning, cognitive response to persuasion, and attitude change. In: Psychological Foundations of Attitudes, Academic Press Inc., New York (1968) 147–169
30. Grimm, S., Motik, B.: Closed world reasoning in the semantic web through epistemic operators. In: Proceedings of the Workshop OWL – Experiences and Directions, CEUR-WS (2005)
31. Patel-Schneider, P.F., Horrocks, I.: Position paper: A comparison of two modelling paradigms in the semantic web. In: Proceedings of WWW2006, ACM Press, NYC, USA (2006) 3–12
32. Stanfill, C., Waltz, D.: Toward memory-based reasoning. Communications of the ACM 29(12) (1986) 1213–1228
33. Kokinov, B.N., Petrov, A.: Integrating memory and reasoning in analogy-making: The AMBR model. In: The Analogical Mind: Perspectives from Cognitive Science, MIT Press, Cambridge, MA (2001) 59–124
34. Kleinberg, J.: Authoritative sources in a hyperlinked environment. Journal of the ACM 46(5) (1999) 604–632
35. Zilberstein, S.: Using anytime algorithms in intelligent systems. AI Magazine 17(3) (1996) 73–83
36. Nováček, V.: Towards an efficient knowledge-based publication data exploitation: An oncological literature search scenario. Technical Report DERI-TR-2009-03-23, DERI, NUIG (2009) Available at http://tinyurl.com/csh3rf. 13 Jul 2010
37. Manola, F., Miller, E.: RDF Primer. (2004) Available at (November 2008): http://www.w3.org/TR/rdf-primer/. 13 Jul 2010
38. Groza, T., Möller, K., Handschuh, S., Trif, D., Decker, S.: SALT: Weaving the claim web. In: ISWC 2007, Busan, Korea (2007)
39. Maedche, A., Staab, S.: Discovering conceptual relations from text. In: Proceedings of ECAI 2000, IOS Press, Amsterdam, Netherlands (2000)
40. Blaschke, C., Andrade, M., Ouzounis, C., Valencia, A.: Automatic extraction of biological information from scientific text: Protein-protein interactions. In: Proc. Int Conf Intell Syst Mol Biol, Protein Design Group, CNB-CSIC, Madrid, Spain (1999) 60–67
41. Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. MIT Press, Cambridge, MA (1998)
42. Cimiano, P., Pivk, A., Schmidt-Thieme, L., Staab, S.: Learning taxonomic relations from heterogenous sources of evidence. In: Buitelaar, P., Cimiano, P., Magnini, B. (eds.) Ontology Learning from Text: Methods, Evaluation and Applications. IOS Press, Amsterdam, Netherlands (2005) 59–73
43. Voelker, J., Vrandecic, D., Sure, Y., Hotho, A.: Learning disjointness. In: Proceedings of ESWC'07, Springer, New York (2007)
44. Gärdenfors, P.: Conceptual Spaces: The Geometry of Thought. MIT Press, Cambridge, MA (2000)
45. Aisbett, J., Gibbon, G.: A general formulation of conceptual spaces as a meso level representation. Artificial Intelligence 133(1–2) (2001) 189–232

46. Smolensky, P., Legendre, G.: The Harmonic Mind: From Neural Computation to Optimality – Theoretic Grammar. MIT Press, Cambridge, MA (2006)
47. Sowa, J.F., Majumdar, A.K.: Analogical reasoning. In: Proceedings of ICCS'03. Springer, Berlin, Heidelberg (2003)
48. Sowa, J.F.: A dynamic theory of ontology. In: Proceedings of FOIS'06, IOS Press, Amsterdam, Netherlands (2006)
49. Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F., Stein, L.A.: OWL Web Ontology Language Reference. (2004) Available at (February 2006): http://www.w3.org/TR/owl-ref/. 13 Jul 2010
50. ter Horst, H.J.: Completeness, decidability and complexity of entailment for rdf schema and a semantic extension involving the owl vocabulary. Journal of Web Semantics 3(2-3) (2005) 79–115
51. Motik, B., Grau, B.C., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C.: OWL 2 Web Ontology Language: Profiles. Working draft, available at http://www.w3.org/TR/owl2-profiles as of Dec 11 (2008). 13 Jul 2010
52. Noy, N., Rector, A.: Defining N-ary Relations on the Semantic Web (2006). Available at (June 2008): http://www.w3.org/TR/swbp-n-aryRelations/. 13 Jul 2010
53. Laskey, K.J., Laskey, K.B., Costa, P.C.G., Kokar, M.M., Martin, T., Lukasiewicz, T.: Uncertainty Reasoning for the World Wide Web. (2008) W3C Incubator Group final report, available at http://www.w3.org/2005/Incubator/urw3/XGR-urw3-20080331/ as of Dec 11, 2008. 13 Jul 2010