Chapter 9
Lightweight Ontologies

John Davies

9.1 Introduction

Ontology was, in its original sense, a branch of philosophy and in this sense of the word is the study of what (kinds of) things exist: it seeks to identify (or posit) the categories of existence and the relationships between those categories; and to define or describe entities[1] within this framework. As such, the ultimate goal of ontology is to provide a definitive and complete classification of entities and the relations between those entities in all spheres of reality, both material and abstract. In this context, the phrase "lightweight ontology" has little meaning: indeed, the purpose of the enterprise is to be as "heavyweight" as to define a framework which is as complete and definitive a representation of reality as possible.

The term ontology has also, however, come to prominence over recent years in computer science and in particular in the areas of knowledge representation (KR) and reasoning and semantic web technology.

As described in more detail in Smith and Welty (2001), interest in ontology arose in 3 separate but related areas of computer science. At a high level, this interest arises from the realisation that the behaviour of a software system is meaningful only by virtue of the interpretation put on it by users of the system. As such, a system which employs a more accurate model of the user's world will in general be more meaningful to that user and will be better able to accommodate change.

In the area of database management systems, it was found that, as database technology itself matured and stabilised, the issue of conceptual modelling was a more important and subtle one: the quality of a requirements specification and ultimately that of the resulting information system itself turned out to be heavily dependent on the ability of a developer to extract and represent accurately knowledge about the modelled domain.

J. Davies (✉)
BT Innovate, British Telecommunications plc, London, UK
e-mail: john.nj.davies@bt.com

[1] We define an entity to be that which has a distinct separate existence.

R. Poli et al. (eds.), *Theory and Applications of Ontology: Computer Applications*,
DOI 10.1007/978-90-481-8847-5_9,

Similarly, in software engineering the 1980s saw the emergence of object-oriented technologies which encouraged an approach to software development requiring the modelling of the application domain using (classes of) objects which could have associated with them both data and "methods" – software procedures which in some sense model an object's capabilities. Objects can invoke the methods of other objects via a message-passing protocol. Programs are then seen as the behaviour of a set of cooperating objects rather than the traditional set of instructions to the computer. In such a paradigm, the issue of a formal and consistent basis for "object modelling" quickly arose.

In artificial intelligence (AI), the attempt to explicitly represent and reason about "knowledge" led to recognition of the relevance of work done in ontology. AI practitioners created knowledge bases representing ground facts and axioms about some domain, typically accompanied by a more or less formal mechanism for automated reasoning, allowing the derivation of new information in a specific instance. McCarthy (1980) stated that creators of intelligent systems must "list everything that exists, building an ontology of our world."

Thus the KR community and, latterly, the semantic technology community began to use the term "ontology" in a somewhat different sense to its original meaning. An ontology increasingly came to mean a model of a domain of interest described in a logical language. Of course, for a software system, that which "exists" is precisely that which can be represented and processed (reasoned about) and so this focus on *logic-based* descriptions is to some extent natural. Logics provide a proof theory (a set of axioms and inference rules which can be used to make deductions) and a model theory (a formal analysis of the relationship between statements in the logic and the world being represented). Indeed, it can be argued that knowledge representation languages which do not have semantic models of the type furnished by logic-based approaches do not actually represent anything: without a clear account of what statements in the language claim to be true in the world, how are we to know what is being represented? Non-logical presentations only find their meaning when processed by computer programs: and different programs may process the same data differently because the lack of an explicit semantic model can lead to different (implicit) semantics being ascribed by different programs.

In addition to a clear semantic account, the use of logic offers other advantages including a clear understanding of the consistency and decidability of the logic at hand. Logics can be shown to be consistent: that is, providing the information represented in the logic initially is itself consistent, no inconsistency can arise from the application of the logical inference rules. Decidability is particularly important in the computational context, concerning as it does the tractability of theorems provers. A decidable logic is one wherein a theorem prover if given a statement to prove will be able to determine in a finite time whether or not the statement is true or false. Other interesting properties of logics are soundness and completeness which inform us about the relationship between the proof theory and model theory of a logic. If a logic is sound, all formulae which can be derived from a set of formulae F using the proof theory are true in all models in which F are true. If a logic is complete, everything true in a given model is provable by the corresponding proof theory.

The availability of these kinds of mathematical results about logics are invaluable in telling us the computational properties of systems we may implement based on them. Although these results are sometimes negative (as when a logic is incomplete or undecidable), the key point is that the results are known at all: for more ad-hoc representation schemes no such results are typically known. McDermott (1978) argues convincingly that these properties are important from more than a purely theoretical point of view. In essence, his point is that it is crucial that a system not only "works" but is *understood*.[2]

However, if we move away from the requirement that ontologies be machine-processable the picture is a little different. In the case where an ontology is used in order to improve the communication between different departments in a company, for example, a formal approach may not be necessary. Indeed, such an approach may be actually unhelpful: the majority of users of such an ontology are likely to have a greater shared understanding of an ontology expressed in a natural language than one expressed in a description logic, for example. Uschold (Building Ontologies: Towards a Unified Methodology, 1996) makes the distinction between the goal of an ontology and the need for formality. He suggests that when it is used to improve the communication between people, natural language can be sufficient. Our focus in this chapter, however, will be on *machine-processable* ontologies and, as argued above, therefore on *logic-based* approaches.

Smith (2003) discusses the relationship between ontology in its older philosophical sense and its later usage in computer science in more detail. For the purposes of this chapter, we focus on the meaning of the term as used by computer scientists, noting that, as mentioned above, this is the context in which the term "lightweight ontology" is meaningful.

The most commonly quoted definition of this sense of the term is that given by Gruber (1992, 1993):

> An ontology is an explicit specification of a conceptualisation

In this sense, the word refers to a *software artefact*: a computer-processable model of some domain of interest. *Lightweight ontologies* in this context are such providing the simplest formalization of the simplest model, adequate for the task at hand. The rationale is that simple ontologies are often more appropriate and economical; they are easier to understand, adapt, management, update, and use. Lightweight ontologies can "survive" in computationally extreme environments where scale and performance are critical, e.g. very large databases and search engines.

In the next section, we consider this notion of ontology in a little more detail, before proceeding to distinguish lightweight ontologies from other ontologies. We

[2]Note that while advocating a logic-based approach we are not advocating any specific logic (such as description logic, for example). Indeed, currently available logics lack some of the properties which would seem to be required for representing and reasoning at the scale and lack of precision found on the web. [Fensel and van Harmelen 2007] discuss new, more appropriate inference mechanisms.

then motivate and describe techniques which have been developed to translate other schemes for information modelling into lightweight ontologies. In Section 9.3, we then consider the Semantic Web and its ontological languages in more detail before proceeding over the next 3 sections to discuss the application of ontologies to information integration, knowledge management and service-oriented environments. Section 9.7 contains some brief concluding remarks.

9.2 Lightweight Ontologies

9.2.1 Lightweight Ontologies and the Semantic Spectrum

We mentioned in the Introduction that ontologies were described in some logical language with an associated underlying semantics and certainly this was the view of early proponents of the application of philosophical ontology to AI (McCarthy, 1980). However, by 1999 it was apparent that the use of the term ontology in the AI community was proliferating and being applied more and more widely. A panel at that year's AAAI conference[3] reported on a wide spectrum of structuring mechanisms that had been characterised as ontologies. Figure 9.1 shows our own adaptation of that spectrum in the light of more recent developments.

Figure 9.1 depicts a number of approaches to information modelling in roughly increasing order of expressiveness which have been adopted in recent years, many of which have attracted the description "ontology." Note that Obrst in Chapter 2 offers a similar model.

By *term list* is meant any set of terms used to denote entities in a particular domain of interest. A good contemporary example of a term list would be a tag cloud. In Web 2.0[4] parlance, a tag is a (relevant) keyword or term associated with or assigned to a piece of content, thus describing the item and enabling keyword-based classification of information for the purpose of browsing and retrieval. A tag cloud is then a set of tags defined on a particular body of content to enable topic browsing. Each tag in the tag cloud is a link to the collection of items that have that tag. Tags are usually chosen by the author or consumer of the content and are thus not part of any shared, more formal classification scheme. Tag clouds are sometimes

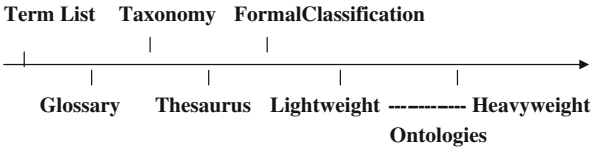

Fig. 9.1 Semantic spectrum

[3] http://www.cs.vassar.edu/faculty/welty/aaai-99/

[4] http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html

also referred to as "folksonomies" (from "folk" + "taxonomy", though folksonomies have little in common with taxonomies, being devoid of structure). Glossaries are closely related to term lists, being term lists with associated definitions for each term.

Thesaurus is a term which, as seen earlier in the case of the term ontology, has been adapted in computer science and taken on a related but different meaning from its original sense. Originally denoting a book listing words in groups of synonyms (and sometimes also antonyms and related concepts), in computer science thesaurus typically denotes a collection of terms denoted by three relations: broader-than (BT), narrower-than (NT) and related-term (RT) sometimes augmented by further relations. Thesauri in this context are used by electronic information providers to associate terms with documents, often in digital libraries, to assist information retrieval and browsing. The semantics of thesauri relations are not always entirely clear, as we will discuss later.

A *taxonomy* is semantically more rigorous than a thesaurus in that it is comprised of a hierarchy of concepts linked by a transitive subsumption relation (often called **isA** or **subClassOf**) whereby each instance of a class can be inferred to be an instance of all parent classes. Taxonomies are strict hierarchies: each class has at most one parent.[5]

Formal classification schemes have recently been proposed (Giunchiglia et al., 2005) in an attempt to formalise previous work on classification. Classification has a long history as the discipline of grouping related concepts or entities. Its task is to aggregate items for some specific purpose, in contrast to ontology, where the goal is generate a model of some domain of interest. Giunchiglia (2005) describes the representation of classifications as lightweight ontologies, as discussed later.

As mentioned earlier, ontologies have been defined, in our sense of the word, as "specifications of conceptualisations." Borst (1997) extended this definition somewhat:

> An ontology is a formal, explicit, specification of a shared conceptualisation.

Similarly, in perhaps the best definition we have encountered, Guarino and Giaretta (1995) offered the following:

> a logical theory which gives an explicit, partial account of a conceptualisation.

This definition emphasises the requirement for a logical theory and that any such account will always be partial rather than being able to completely specify the intended meaning of any conceptual element.

Our own view is that to qualify as such, an ontology should offer a formal semantic account of statements in the ontological language: that is, they should be *logic-based*.[6] Indeed, without formal semantics, it is hard to see how an ontology

[5]Taxonomy is also sometimes used loosely (and incorrectly) to denote any set of categories against which electronic content has been classified.

[6]Though note that we do not necessarily exclude non-logical but formal accounts (e.g. in the case where the knowledge is probabilistic in nature).

can be sharable or re-usable, since there is no clear account of what statements in the language used to represent the ontology actually mean. In addition, the representation should be *declarative* so that ontological concepts and other features (such as the constraints imposed on their use) are explicitly defined. In addition, in order to qualify as a *conceptualisation* an ontology must be more than a set of ground facts[7]: rather it is an abstract model of concepts in the world, usually limited to a particular domain of interest.

Lightweight ontologies would then typically consist of a hierarchy of concepts and a set of relations holding between those concepts. As discussed earlier, if we specify only a hierarchy of concepts related using a subsumption relation, then we have a *taxonomy*. Conversely, *heavyweight* ontologies add cardinality constraints, standalone axioms, reified statements and more.

Depending on the planned use of the ontology, a deeper[8] ontology may or may not be required. The deeper an ontology, the more resources it requires to construct and maintain and of course justifying the required investment in ontology creation will depend on the anticipated value of the uses(s) to which the ontology will be put.

We stated above that ontologies should be specified in a logical language and much research in recent years has focussed on the use of Description Logics for this purpose. Description Logics (DLs) are logics based on a subset of first order predicate calculus, so called because they focus on descriptions of concepts (classes) as a principal means for expressing logical propositions. A description logic system emphasises the use of classification and subsumption reasoning as its primary mode of inference. DLs were designed as an extension to frames and semantic networks, which were not equipped with formal logic-based semantics. DLs are a very natural candidate for specifying ontologies, given their focus on concepts and subsumption reasoning and the fact that, being subsets of FOPC, they have attractive computational properties. Indeed, there has been extensive research into the theoretical underpinnings of DLs and a family of logics set out, each with well-understood properties in terms of expressive power and computational tractability (Baader et al., 2003).

In recent years, the role and impact of ontologies in computer science has increased significantly with the advent of the semantic web (Fensel, 2001). In the semantic web, ontologies provide the key mechanism whereby web-based metadata is made machine-interpretable. They provide the formal, shared, explicit domain model against which web data can be annotated. The World Wide Web Consortium (W3C) has developed the OWL family of languages as open standards for specifying and using ontologies on the (semantic) web. In Section 9.6 we review the OWL variants and their relative expressivity and complexity.

In recent years, a number of researchers have constructed lightweight ontologies for use in a number of different application scenarios, often providing mechanisms

[7] The OWL web ontology language, for example, allows a set of ground facts to be defined as an ontology. Strictly speaking, this is inadmissible.

[8] 'Deeper' in the sense more elaborate and offering a more precise model of the domain at hand.

to map between other formalisms on the semantic spectrum into a lightweight ontology as we have defined it. This work is important since, firstly, it adds to these previous schemes the benefits of rigour, formal semantics and improved machine processability and, secondly, because it provides a way of generating domain ontologies at reasonable cost based on pre-existing work typically with at least some degree of community consensus.

In the remainder of this section, we will give examples of methods of converting schemes from a number of points lower on our semantic spectrum, before proceeding in the rest of the chapter for a more detailed discussion of lightweight ontologies on the semantic web.

9.2.2 Folksonomies and Lightweight Ontologies

Motivated by the wide use of folksonomies on the one hand and the benefits of the more formal, consistent ontological approach on the other, Van Damme et al. (2007, 2008) describe a method for "turning folksonomies into ontologies."

As described above, aggregation of raw user-supplied metadata (tags) leads to a tag cloud or folksonomy, as exemplified in systems such as Flickr[9] or deli.cio.us.[10] Problems with the use of unstructured, uncontrolled tag clouds include:

(i) different tags referring to the same concept ("Mr Bush" "George Bush" "the president");
(ii) the same tag referring to different concepts ("bank" referring to a financial institution or referring to an area of sloping land, for example along a riverside);
(iii) different users tagging the same content at different conceptual levels of abstraction ("Eiffel Tower" or "Paris").

Building on previous work including Specia and Motta (2007), Van Damme and her co-authors propose five sets of resources available for deriving ontologies from folksonomies:

(i) the statistical analysis of both folksonomies and the usage of folksonomy-based systems (including the underlying social relationships between users) to identify structural patterns in folksonomies;
(ii) available lexical resources such as dictionaries, Wordnet and Wikipedia;
(iii) existing ontologies
(iv) tools for ontology mapping and matching
(v) methodologies for assisting a community (of ontology-builders in a given domain) in finding and maintaining consensus.

The contribution of each of the above type of resource to the process is discussed and analysed.

---

[9] http://www.flickr.com/
[10] http://del.icio.us/

9.2.3 Thesauri and Lightweight Ontologies

Thesauri are controlled vocabularies developed in specific domains for the purpose of annotating electronic content and exploiting such annotations for enhanced information retrieval and browsing. Most thesauri are represented in special purpose data formats and often use relations between categories whose semantics are not well-defined. Thus converting thesauri to lightweight ontologies, as is in the case folksonomies, brings a number of benefits:

- adherence to a open standard and the consequent enhanced interoperability
- a formal semantic basis
- an explicit specification
- a machine-processable representation

Assem et al. (2006) present a method for the conversion of thesauri to a lightweight ontology SKOS (Miles and Brickley, 2005). Essentially, SKOS is used as a metamodel for representing thesauri in RDF. Three specific cases are analysed, revealing that two of the cases have non-standard features not (currently) anticipated by the method presented. Nevertheless, it is concluded that the metamodel does seem applicable for representing resources adhering to the ISO2788 standard for monolingual thesauri.

9.2.4 Formal Classification and Lightweight Ontologies

As mentioned above Giunchigla et al. (2005) introduces the notion of *formal classification*. Noting that human-crafted classifications (e.g. DMOZ,[11] Dewey Decimal Classification[12]) lack a key ontological property, namely representation in a formal language over which automated reasoning can be performed, formal classification is developed as a graph structure where labels are written in a formal concept language. It is shown that formal classifications are equivalent to a form of lightweight ontology. The notion of Normalized Formal Classification (NFC) is developed. An NFC is an FC wherein labels of child nodes are always more specific than the labels of their parent nodes. A fully automated method is then presented for document classification into NFCs using propositional reasoning.

9.3 Ontologies and the Semantic Web

The principal application area for lightweight ontologies in Computer Science today is in the application of semantic technology to a number of application areas. This smenatic technology has been developed as part of the W3C's semantic web

[11] http://www.dmoz.org/
[12] http://www.oclc.org/dewey/

initiative, which is described in this section. The term semantic technology is a broader term, used to denote the application of semantic web technology both on the web and in other areas.

Despite its explosive growth over the last decade, the Web remains essentially a tool to allow humans to access information. The next generation of the web, dubbed "the Semantic Web," will extend the web's capability through the increased availability *machine-processable* information.

Currently, web-based information is based primarily on documents written in HTML, a language useful for describing the visual presentation of webpages through a browser. HTML and today's web, however, offer only very limited ways of describing the *content* itself. So, for example, you can specify that a given string should be displayed in a large bold font but you cannot specify that the string represents a product code or product price.

Semantic Web technology aims to address this shortcoming, using the descriptive languages RDF and OWL, and the data-centric, customizable markup language XML. These technologies, which are standards of the W3C[13] (WorldWideWeb Consortium), allow rich descriptions of the content of Web documents. These machine-processable descriptions in turn allow more intelligent software systems to be written, automating the analysis and exploitation of web-based information.

In this paper, we begin by describing the key technology building blocks of the semantic web, namely the languages XML and RDF and the notion of ontologies. We then proceed to discuss the application of this technology in the key application area of eBusiness through the use of semantic web services.

Underpinning the Semantic Web is a stack of languages, often drawn in a Fig. 9.2 first presented by Berners-Lee in a presentation to the XML 2000 conference[14]:

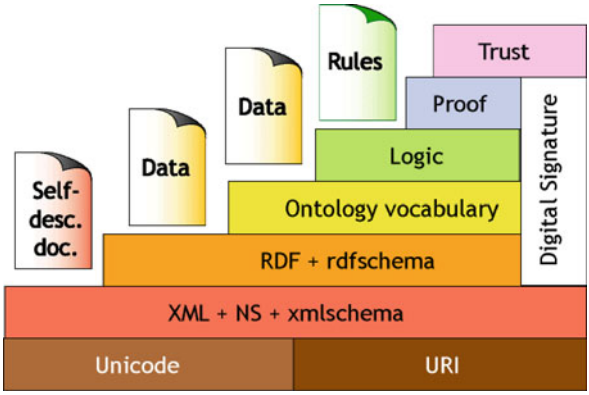

Fig. 9.2 Semantic web technology layers

[13] http://www.w3c.org/

[14] http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide1-0.html

We will briefly discuss the XML, RDF and Ontology vocabulary layers in this language-stack before describing ontologies and their use in the semantic web.

XML is already a widely-used language designed for annotating documents of arbitrary structure with information concerning the *content* of those documents, as opposed to HTML, which was designed principally for information *presentation* as described in the introduction. A well-formed XML document creates a balanced tree of nested sets of open and close tags. There is no fixed tag vocabulary or set of allowable combinations, so these can be defined for each application. One set of XML documents might use tags such as <price> and <quantity> and <delivery-date>, while another might use <author>, <title> and <abstract>.

Perhaps the most important use of XML is as a uniform data-exchange format. An XML document can essentially be transferred as a data object between two applications.

As mentioned above, XML is a useful language for attempting to define data exchange formats, particularly when building new interoperable systems from scratch. However, new systems will frequently need to interact with pre-existing systems. Different pre-existing systems will very often use the same term for different concepts. These types of conflicts typically require more extensive *semantics-based* solutions.

Figure 9.3 below shows two examples of semantic conflicts that can found across data sets. These conflicts are very frequent, occurring as a natural consequence of data modeling – whether due to isolated development, changing needs, organizational or structural differences, or simply the different approach of 2 human data modellers.

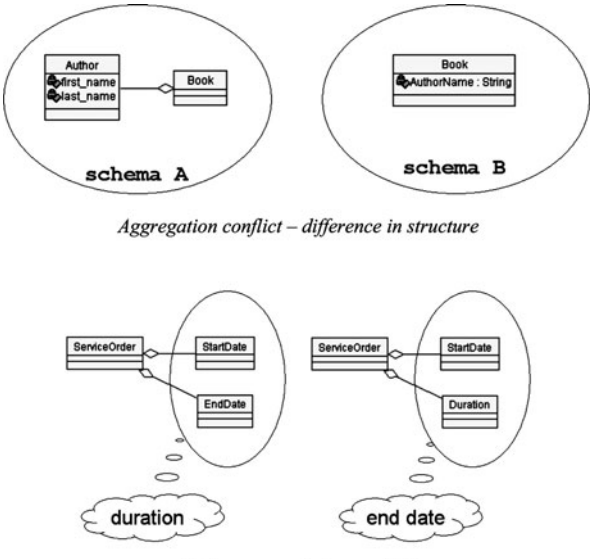

Fig. 9.3 Types of semantic conflicts (adapted from Pollock and Hodgson, 2004)

It is apparent that XML only partially addresses the data interoperability issue. We need interoperation between the processes and information sets within and across organizations without costly point-to-point data and terminology mappings. The goal of semantic web technologies is to employ logical languages that expose the structures and meanings of data more explicitly, thereby allowing software to interact whether terms and definitions are equivalent, different, or even contradictory. This is the goal of the RDF and OWL languages.

XML can be seen as *prescriptive*, in that it pre-defines the format of a data object, RDF is *descriptive*, allowing the description of semantic relationships between web resources, through the use of <subject, verb, object> triples. Sets of these triples can also be viewed as creating a graph structure. This graph structure in effect creates a web of *meaning*, where links between web resources have a semantic element (in contrast to today's web, where pages may be linked but it is for a human user to interpret why they are linked and the relationship between the information they contain).

For example, the graph structure in Fig. 9.4 expresses the following three triples:

1. "http://www.famousauthor.org/id21", hasName, "J Tolkien">
2. "http://www.famousauthor.org/id21", authorOf, "http://www.books.org/ISBN00615861"
3. "http://www.books.org/ISBN00615861", hasPrice, "£39">

Notice how the RDF triples allow us to combine references to web resources with literal values (e.g. character strings or numbers). Triple 1 above, for example, roughly speaking states that the famous author described at web location "http://www.famousauthor.org/id21" has the name "J Tolkien".

RDF[15] allows us to build a concept map (or ontology) of our domain, defining the key classes or concepts (authors, books, etc) and the relationships between those concepts (e.g. authors write books). RDF then allows us to represent and reason about specific instances of those concepts (e.g. "J. Tolkien wrote the book described at http://www.books.org/ISBN00615861").

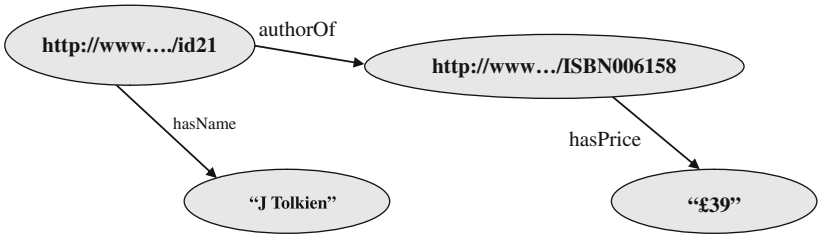

Fig. 9.4 RDF triples as a graph

[15] Strictly, RDF and its sister language RDF Schema

Crucially, RDF Schema includes a number of primitives which have precisely defined semantics. This means that, unlike in XML, the meaning of these primitives are formally defined and hence known to any application.

The OWL Web Ontology Language[16] is a language for the "Ontology vocabulary layer of the semantic web layer cake (Fig. 9.2) and is, like XML and RDF, a standard of the W3C. Ontologies written in OWL facilitate greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with a formal semantics. OWL has three increasingly-expressive sublanguages: OWL Lite, OWL DL, and OWL Full.

In a nutshell, RDF and OWL provide a framework for establishing relationships between data and for specifying constraints on different data elements and their relationship to one another. Furthermore, they allow for a description of pre-existing web-based data rather than prescribing a rigid format for web pages as in XML. This means that data on the web becomes machine-processable as well as human readable. In turn, this allows machines to do more of the hard work of interpreting data which today is left to the user.

In the next sections, we discuss the applications of semantic technology (and hence lightweight ontologies) to information integration, knowledge management, and to service-oriented environments.

9.4 Ontologies and Information Integration

Modern organisations are characterised by the availability of massive volumes of information, made possible by electronic technology. The ability to find and share information rapidly and effectively is a clear commercial advantage in all sectors. So, too, is the ability to retain information. Maintaining the corporate memory bank as employees leave the company is an oft discussed issue.

To these commercial issues have been added regulatory ones. Organisations which do not disclose all relevant information to regulatory authorities may be seriously penalised. Yet the organisation can only disclose information it knows it has. Information lost on corporate computers can not be disclosed at the appropriate time; but will certainly be revealed if the organisation is subject to a detailed forensic analysis of hard drives prior to a legal hearing.

A typical large organisation in an information-intensive sector (such as finance) will have a number of data silos (e.g. an HR system, a CRM system, one or more billing systems, etc). Each such system has its own data model, no two of which are typically the same, making exchange and integration of information difficult: indeed, analysts report that the majority of businesses resort to expensive new software and/or manual processes when confronted with a need to integrate information from multiple sources.

[16] http://www.w3.org/TR/owl-guide/

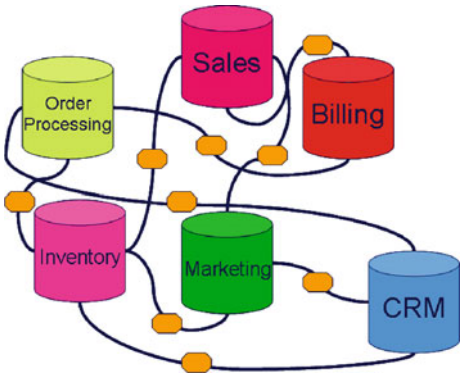

Fig. 9.5 Information integration

The value of ontologies in information integration stems from the ability to create an over-arching ontology which can subsume multiple database schemas. The current state-of-the-art in information integration is represented in Fig. 9.5. To achieve integration at the semantic level, mappings are created between each database. These might be databases internal to one organisation, e.g. order processing and stock control databases; or the mappings might be across organisations, e.g. between databases held by separate companies working together in a joint venture or supply chain. In any case, the problem is that the number of mappings increases quadratically with the number of databases.

Ontologies can help to address this issue by providing a uniform access layer to heterogeneous data sources: the linkage of multiple structured, semi-structured and unstructured information sources using a consistent vocabulary makes it easier to build applications pulling data together from across the enterprise and also facilitates the introduction of new systems and databases (Fig. 9.6).

Advantages of the semantic integration approach as opposed to others include:

- no need to re-engineer legacy data sources – existing data sources are instead "wrapped" by a semantic description;
- based on lightweight, open standards from W3C;
- inherently extensible – RDF and OWL have been designed to make it relatively straightforward to integrate concepts and relations from more than one ontology;

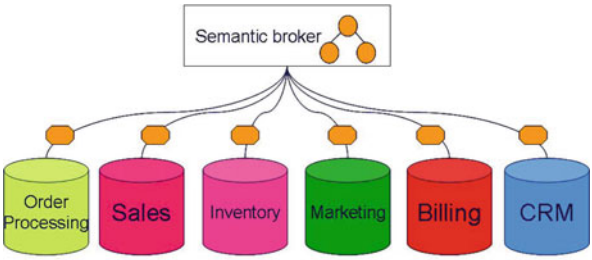

Fig. 9.6 Semantic information integration

- reasoning with the information – because OWL (for example) is a logical language, formal reasoning is supported, allowing the inference of new facts from the explicitly stored information and allowing the definition of business rules.

We have reduced the number of mappings needed, but they still do have to be created. One way to create mappings is to use a mapping language. This is fine for specialist knowledge engineers but others need a more natural and intuitive approach which is easy to learn and use. A number of graphical mapping tools have been created for such users. One such has been developed as part of their OntoStudio ontology engineering environment.[17]

Simple drag-and-drop functionality is used to create and amend mappings. At the same time, the system undertakes consistency checks to ensure that the user's actions make sense. Figure 9.7 shows a view of the mapping tool. The left and right-hand side shows portions of two different ontologies, and the mappings are represented by lines between them. Mappings can even be conditional. Consider,

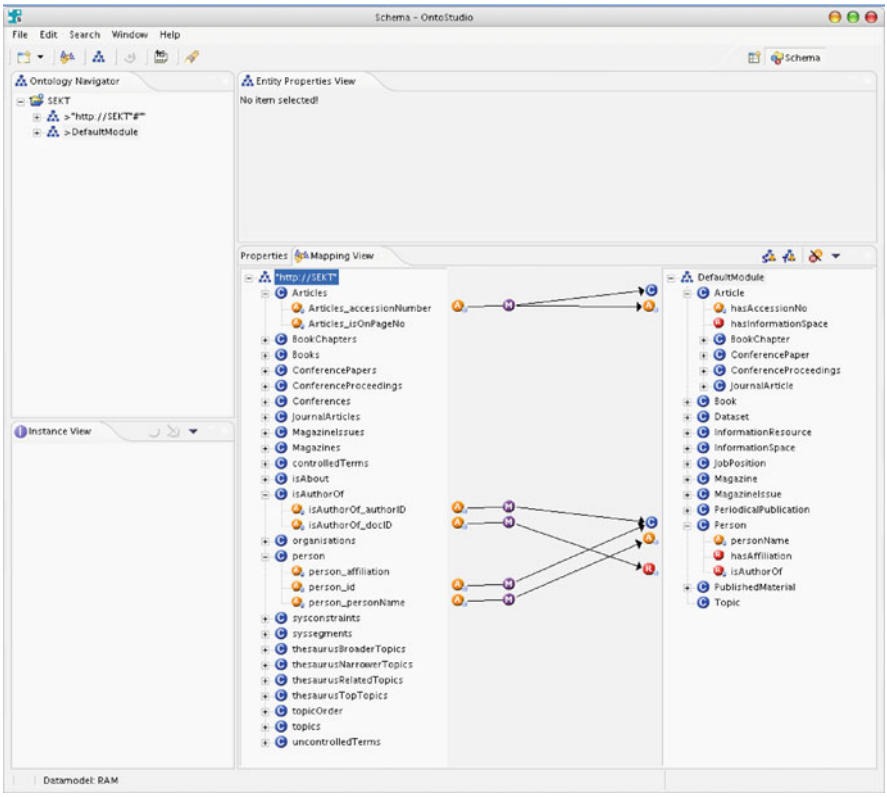

Fig. 9.7 Ontology mapping tool

[17] http://www.ontoprise.de/

for example, a mapping between two national transport ontologies. The definition of a "truck" differs in different countries, depending in some countries on the weight of the vehicle. This can be taken into account when creating the mapping.

Even greater gains can be achieved by automating, at least partially, the process of creating the mappings. This is an area of current research. A starting approach is to look for similarities in the text strings used to denote data fields by different schemas, e.g. *phone* for *telephone*. We can even take account of different representations of similar sounds, e.g. the use of *4* to represent *for*. This is frequently called syntactic matching. We can introduce some appreciation of semantics by using a thesaurus, such as Wordnet,[18] to identify synonyms. Semantic matching can go further by taking account of the structure inherent in two schemas. For example, a product classification system can in general be represented as a graph.[19] Structural similarities then enable the software to draw reasonable conclusions about the relationship between nodes (i.e. categories of products) in two classification systems. The software may propose equivalences between categories, or that a category in one system is a subset of a category in the other classification. Readers interested in the technical detail of one approach, based on the use of a form of logic known as propositional calculus, are referred to Bouquet et al. (2003).

Once these techniques have been used to create an initial mapping, it can then be loaded into a graphical editing tool and refined manually.

The end result is that it is possible to integrate heterogeneous databases, and provide the knowledge worker in an organisation with a unified view across these databases.

9.5 Ontologies and Knowledge Management

We have seen in the previous section how semantic technology can be applied to the integration of structured information. At least as pressing an issue, and more technically challenging, is management of *unstructured* information, part of the knowledge management problem.[20]

[18] Wordnet is a lexical reference system in which English nouns, verbs, adjectives and adverbs are organised into synonym sets, with relations linking the sets. Wordnet provides both a web-based user interface for the casual user and also a programming interface to enable incorporation into other systems, e.g. software for mapping between different terminologies.

[19] In general a graph, but frequently a tree where the product classification is organised as a strict hierarchy.

[20] It is not our intention to discuss the definition of knowledge management here but a broad definition could be "the management by an organisation of its intellectual assets to deliver more efficient and effective ways of working." Better access to and management of unstructured information is a key part of this endeavour.

9.5.1 Limitations of Current Technology

The traditional approach is to provide tools (in particular search engines) based on text-string matching. This may be simply through a user initiating a search, or through text searches embedded in an application. In any case, there are several problems with this approach, which can be divided into four main areas:

(i) *Query Construction*

In general, when specifying a search, users enter a small number of terms in the query. Yet the query describes the information need, and is commonly based on the words that people expect to occur in the types of document they seek. This gives rise to a fundamental problem, in that not all documents will use the same words to refer to the same concept. Therefore, not all the documents that discuss the concept will be retrieved by a simple keyword-based search. Furthermore, query terms may of course have multiple meanings (query term *polysemy*). As conventional search engines cannot interpret the sense of the user's search, the ambiguity of the query leads to the retrieval of irrelevant information.

Although the problems of query ambiguity can be overcome to some degree, for example by careful choice of additional query terms, there is evidence to suggest that many people may not be prepared to do this. For example, an analysis of the transaction logs of the Excite WWW search engine (Jansen et al., 2000) showed that web search engine queries contain on average 2.2 terms. Comparable user behaviour can also be observed on corporate Intranets. An analysis of the queries submitted to BT's Intranet search engine over a 4-month period between January 2004 and May 2004 showed that 99% of the submitted queries only contained a single phrase and that, on average, each phrase contained 1.82 keywords.

(ii) *Lack of Semantics*

Converse to the problem of polysemy, is the fact that conventional search engines that match query terms against a keyword-based index will fail to match relevant information when the keywords used in the query are different from those used in the index, despite having the same meaning (index term *synonymy*). Although this problem can be overcome to some extent through thesaurus-based expansion of the query, the resultant increased level of document recall may result in the search engine returning too many results for the user to be able to process realistically.

In addition to an inability to handle synonymy and polysemy, conventional search engines are unaware of any other semantic links between concepts. Consider for example, the following query:

"telecom company" Europe "John Smith" director

The user might require, for example, documents concerning a telecom company in Europe, a person called John Smith, and a board appointment. Note, however, that a document containing the following sentence would not be returned using conventional search techniques:

> At its meeting on the 10th of May, the board of London-based O2 appointed John Smith as CTO

In order to be able to return this document, the search engine would need to be aware of the following semantic relations:

> O2 is a mobile operator, which is a kind of telecom company;
> London is located in the UK, which is a part of Europe;
> A CTO is a kind of director.

(iii) *Lack of Context*

Many search engines fail to take into consideration aspects of the user's context to help disambiguate their queries. User context would include information such as a person's role, department, experience, interests, project work, etc. A simple search on BT's Intranet demonstrates this. A person working in a particular BT line of business searching for information on their corporate clothing entitlement is presented with numerous irrelevant results if they simply enter the query "corporate clothing". More relevant results are only returned should the user modify their query to include further search terms to indicate the part of the business in which they work. As discussed above, users are in general unwilling to do this.

(iv) *Presentation of Results*

The results returned from a conventional search engine are usually presented to the user as a simple ranked list. The sheer number of results returned from a basic keyword search means that results navigation can be difficult and time consuming. Generally, the user has to make a decision on whether to view the target page based upon information contained in a brief result fragment. A survey of user behaviour on BT's intranet suggests that most users will not view beyond the 10th result in a list of retrieved documents. Only 17% of searches resulted in a user viewing more than the first page of results.[21] Essentially, we would like to move from a document-centric view to a more knowledge-centric one (for example, by presenting the user with a digest of information gleaned from the most relevant results found as has been done in the Squirrel semantic search engine described later in this chapter).

In recent years, considerable effort has been put into the use of ontologies to deal with the problem of managing and accessing unstructured information and we summarise some of the key aspects in the remainder of this section.

[21] Out of a total of 143,726 queries submitted to the search engine, there were 251,192 occasions where a user clicked to view more than the first page of results. Ten results per page are returned by default.

9.5.2 Applying Ontologies in Knowledge Management

The essential challenge is to create some (meaningful) structure out of unstructured text. One way to do this is to create semantic *metadata*: data describing the unstructured text.

Such metadata can exist at two levels. They can provide information about a document or a page, e.g. its author, creation or last amendment date, or subject area (topic); or they can provide information about entities in the document, e.g. the fact that a particular string in the document represents a company, or a person or a product code. The metadata themselves should describe the document or entities within the document in terms of an ontology. At the document level we might have a property in the ontology *has Author* to describe authorship. Within the document we would use classes such as Person, Company or Country to identify specific entities.

Such metadata could be created by the authors of the document. In general this will not happen. The authors of Word documents or emails will not pause to create metadata. We need to generate metadata automatically, or at least semi-automatically.

There are two broad categories of technology which we can use for this: statistical or machine learning techniques[22]; and information extraction techniques based on natural language processing. The former generally operate at the level of documents, by treating each document as a "bag of words". They are, therefore, generally used to create metadata to describe documents. The latter are used to analyse the syntax of a text to create metadata for entities within the text, e.g. to identify entities as Persons, Companies, Countries etc. Nevertheless, this division should not be seen too starkly. For example, one of the goals of the SEKT project (http://www.sekt-project.com) was to identify the synergies which arise when these two different technologies are used closely together. An overview of semantic knowledge management, including these two approaches to creating metadata, is given in Davies et al. (2005). For more detail, see Davies, Studer and Warren (2005), which contains a chapter on each of these approaches, besides information on a number of other topics discussed in this paper.

The metadata can create a link between the textual information in the documents and concepts in the ontology. Metadata can also be used to create a link between the information in the document and instances of the concepts. This process is known as semantic annotation.

To give an example of the linkage between documents and the ontology and knowledgebase, we can imagine that the ontology will contain the concept Company. Then the text string "BT" in the document will be identified as being an instance of the concept Company. This is made possible by natural language

[22] Statistical techniques employ algorithms with well-defined mathematical properties, usually derived based on certain assumptions about the datasets being used. Machine learning techniques are generally heuristic techniques with no formally derived mathematical properties, e.g. iterative techniques for which no proof of convergence exists. The two approaches may suit different circumstances, or can be used together in a complementary fashion.

processing software which can make intelligent deductions about what kind of entities particular text strings represent. For example, the software can make inferences from the way the term "BT" is used, and the words associated with it in the text. If the company British Telecommunications Plc. (to give it its formal name) exists as an instance of Company in the knowledgebase, then the software will tag "BT" as referring to British Telecommunications Plc. This is possible through an understanding of how acronyms are formed. In other situations it may even be able to tag "it" as referring to the same company, on the basis of a relatively shallow analysis of the text.[23]

Where the system identifies a text string as an instance of a concept in the ontology but which is not represented in the knowledgebase, then that instance can be added to the knowledgebase. For example, the text string "ABC Holdings" may be identified as a company, but one not represented in the knowledgebase. The system can then add "ABC Holdings" to the knowledgebase.

Figure 9.8 illustrates part of an ontology and corresponding knowledgebase, and shows how entities in the text can be associated with entities in the knowledgebase.

Research is also in progress to use natural language processing techniques to learn concepts from text, and thereby extend the ontology. However, this is a significantly harder problem. For an example of the state of the art, see Cimiano and Völker (2005).

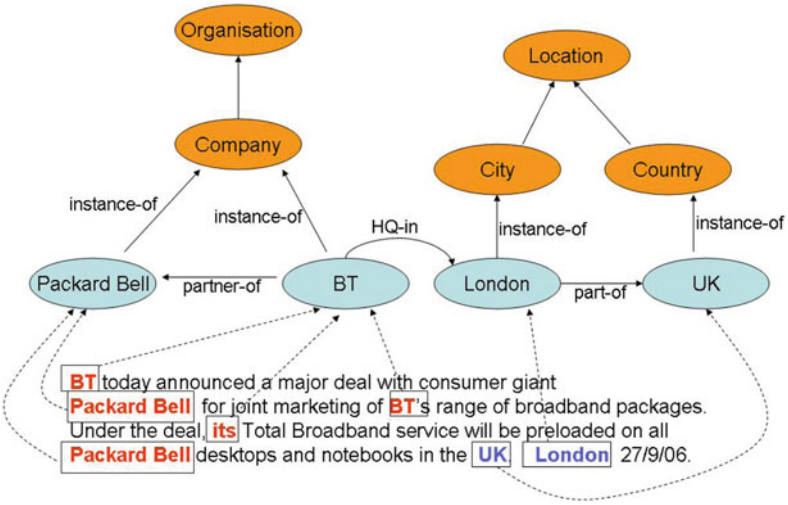

Fig. 9.8 Semantic annotation

[23]The use of pronouns and other short words in place of longer words or phrases is called anaphora. Hence, the matching of such short words with their longer equivalent is called *anaphora resolution*.

9.5.3 Semantic Knowledge Management Tools

The ontological approach offers significant advantages in the capability of tools for the management of unstructured text. In this section, we exemplify with one such tool. Bontcheva et al. (2006) is a more comprehensive discussion of semantic information access discussing a wider range of tools. Similarly Mangrove (2007) contains a survey and classification of ontology-based search technology.

9.5.3.1 Squirrel Semantic Search Engine

Squirrel (Duke et al., 2007) provides combined keyword based and semantic searching. The intention is to provide a balance between the speed and ease of use of simple free text search and the power of semantic search. In addition, the ontological approach provides the user with a rich browsing experience. Squirrel builds on and integrates a number of semantic technology components:

(i) PROTON,[24] a lightweight ontology and world knowledge base is used against which to semantically annotate documents.
(ii) Lucene[25] is used for full-text indexing;
(iii) The KAON2 (Motik and Studer, 2005) ontology management and inference engine provides an API for the management of OWL-DL and an inference engine for answering conjunctive queries expressed using the SPARQL[26] syntax. KAON2 also supports the Description Logic-safe subset of the Semantic Web Rule Language[27] (SWRL). This allows knowledge to be presented against concepts that goes beyond that provided by the structure of the ontology. For example, one of the attributes displayed in the document presentation is "Organisation". This is not an attribute of a document in the PROTON ontology; however, affiliation is an attribute of the Author concept and has the range "Organisation". As a result, a rule was introduced into the ontology to infer that the organisation responsible for a document is the affiliation of its lead author;
(iv) OntoSum (Bontcheva, 2005) a Natural Language Generation (NLG) tool, takes structured data in a knowledge base (ontology and associated instances) as input and produces natural language text, tailored to the presentational context and the target reader. In the context of the semantic web and knowledge management, NLG is required to provide automated documentation of ontologies and knowledge bases and to present structured information in a user-friendly way;
(v) KIM (Popov et al., 2003) is used for massive semantic annotation.

[24] http://proton.semanticweb.org/
[25] http://lucene.apache.org/
[26] http://www.w3.org/TR/rdf-sparql-query/
[27] http://www.w3.org/Submission/SWRL/

Initial Search

Users are permitted to enter terms into a text box to commence their search. This initially simple approach was chosen since users are likely to be comfortable with it due to experience with traditional search engines. Squirrel then calls the Lucene index and KAON2 to identify relevant textual resources or ontological entities, respectively. In addition to instance data, the labels of ontological classes are also indexed. This allows users to discover classes and then discover the corresponding instances and the documents associated with them without knowing the names of any instances e.g. a search for "Airline Industry" would match the "Airline" class in PROTON. Selecting this would then allow the user to browse to instances of the class where they can then navigate to the documents where those instances are mentioned.

Meta-Result

The meta-result page is intended to allow the user to quickly focus their search as required and to disambiguate their query if appropriate. The page presents the different types of result that have been found and how many of each type.

The meta-result for the "home health care" query is shown in Fig. 9.9 under the sub-heading "Matches for your query".

Document View

The user can select a document from the result set, which takes them to a view of the document itself. This shows the meta-data and text associated with the document and also a link to the source page if appropriate – as is the case with web-pages. Semantically annotated text (e.g. recognised entities) are highlighted. A screenshot of the document view is shown in Fig. 9.10.

"Mousing-over" recognised entities provides the user with further information about the entity extracted from the ontology. Clicking on the entity itself takes the user to the entity view.

Fig. 9.9 Meta-result

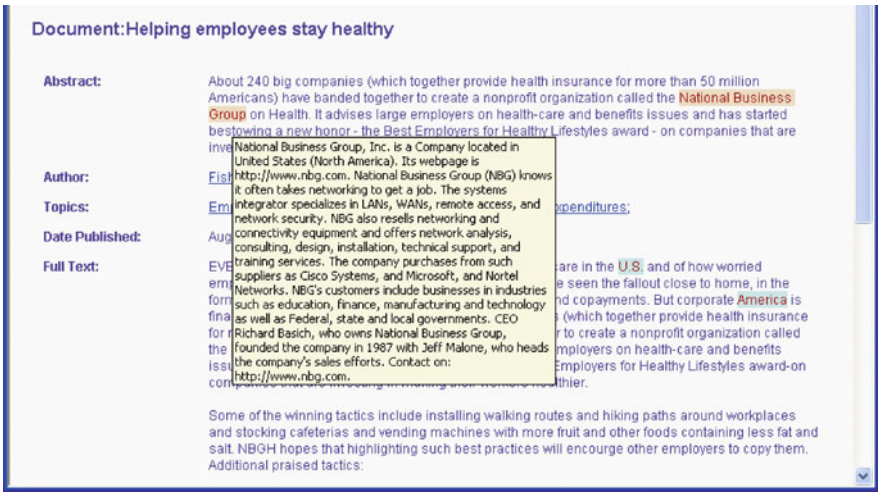

Fig. 9.10 Document view

Entity View

The entity view for "Sun Microsystems" is shown in Fig. 9.11. It includes a summary generated by OntoSum. The summary displays information related not only to the entity itself but also information about related entities such as people who hold job roles with the company. This avoids users having to browse around the various entities in the ontology that hold relevant information about the entity in question.

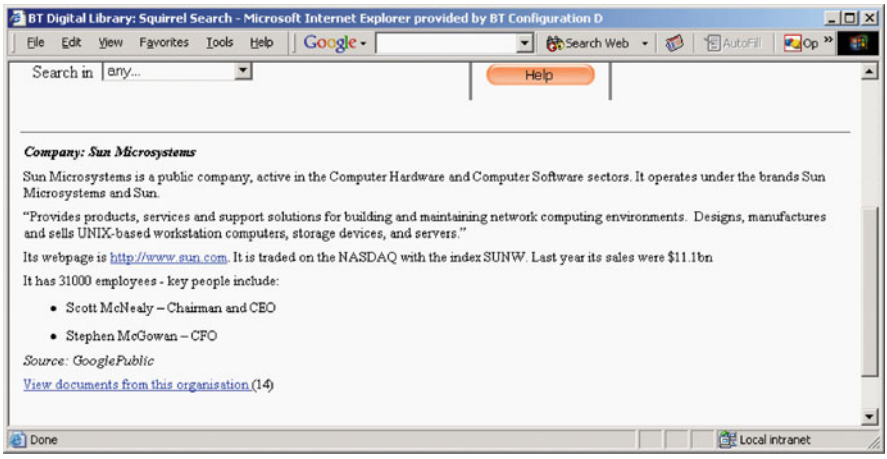

Fig. 9.11 Company entity view

Fig. 9.12 Consolidated results

Consolidated Results

Users can choose to view results as a *consolidated summary* (digest) of the most relevant parts of documents rather than a discrete list of results. The view allows users to read or scan the material without having to navigate to multiple results. Figure 9.12 shows a screenshot of a summary for a query for "Hurricane Katrina". For each subdocument in the summary the user is able to view the title and source of the parent document, the topics into which the subdocument text has been classified or navigate to the full text of the document. Evaluation

Squirrel has been subjected to a three-stage user-centred evaluation process with users of a large Digital Library. Results are promising regarding the perceived information quality (PIQ) of search results obtained by the subjects. From 20 subjects, using a 7 point scale the average (PIQ) using the existing library system was 3.99 vs. an average of 4.47 using Squirrel – an 12% increase. The evaluation also showed that users rate the application positively and believe that it has attractive properties. Further details can be found in Thurlow and Warren (2008).

9.6 Ontologies and Service-Oriented Environments

Industry is seeking urgently to reduce IT costs, more than 30% of which are attributable to integration.[28] Furthermore, in the telecommunications sector for example, costs of OSS[29] integration can rise to 70% of the total OSS budget.[30]

[28] Gartner Group, 2004.

[29] Operational Support Systems: systems that support the daily operation of an organisation's business including, for example, billing, ordering, delivery, customer support.

[30] See, for example, http://www.findarticles.com/p/articles/mi_m0TLC/is_5_36/ai_86708476

In addition, there is a need to reconfigure system components efficiently in order to satisfy regulatory requirements for interoperation and to respond quickly to increasingly sophisticated customer requirements for bundled services.

Thus one of the most pressing current issues design of software architectures is to satisfy increasing software complexity as well as new IT needs, such as the need to respond quickly to new requirements of businesses, the need to continually reduce the cost of IT or the ability to integrate legacy and new emerging business information systems. In the current IT enterprise settings, introducing a new product or service and integrating multiple services and systems present unpredicted costs, delays and difficulty. Existing IT systems consist of a patchwork of legacy products, monolithic off-shelf applications and proprietary integration. It is even today's reality that in many cases users on the "spinning chairs" manually re-enter data from one system to another within the same organization. The past and existing efforts in Enterprise Application Integration (EAI) don't represent successful and flexible solutions. Several studies showed that the EAI projects are lengthy and the majority of these efforts are late and over budget. It is mainly costs, proprietary solutions and tightly-coupled interfaces that make EAI expensive and inflexible.

Service Oriented Architecture (SOA) solutions are the next evolutionary step in software architectures. SOA is an IT architecture in which functions are defined as independent services with well-defined, invocable interfaces. SOA will enable cost-effective integration as well as bring flexibility to business processes. In line with SOA principles, several standards have been developed and are currently emerging in IT environments. In particular, Web Services technology provides means to publish services in a UDDI registry, describing their interfaces using the Web Service Description Language (WSDL) and exchanging requests and messages over a network using SOAP protocol. The Business Process Execution Language (BPEL) allows composition of services into complex processes as well as their execution. Although Web services technologies around UDDI, SOAP and WSDL have added a new value to the current IT environments in regards to the integration of distributed software components using web standards, they cover mainly characteristics of syntactic interoperability. With respect to a large number of services that will exist in IT environments in the inter and intra enterprise integration settings based on SOA, the problems of service discovery or selection of the best services conforming user's needs, as well as resolving heterogeneity in services capabilities and interfaces will again be a lengthy and costly process. For this reason, *machine processable semantics* should be used for describing services in order to allow total or partial automation of tasks such as discovery, selection, composition, mediation, invocation and monitoring of services. As discussed, the way to provide such semantics is through the use of ontologies.

Web services technology effectively added computational objects to the static information of yesterday's Web and as such offers a distributed services capability over a network. Web services provide an easy way to make existing (or indeed new) software components available to applications via the Internet. As explained above, however, web services are essentially described using semi-structured natural language mechanisms, which means that considerable human intervention is needed to

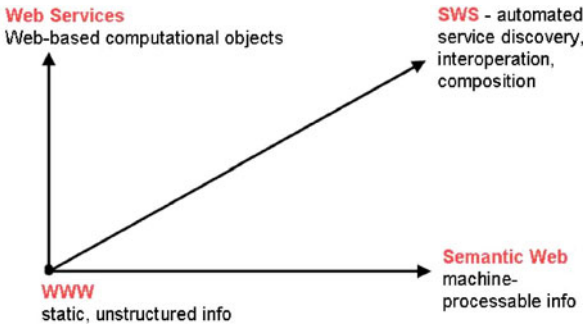

Fig. 9.13 Web services and the semantic web

find and combine web services into an end application. As the number of services available grows, the issue of scalability becomes critical: without richer (machine-processable, semantic) descriptions of services it becomes impossible to discover and compose services in an efficient manner.

The Semantic Web enables the accessing of web resources by semantic descriptions rather than just by keywords. Resources (here, web services) are defined in such a way that they can be automatically processed by machine. This will enable the realisation of Semantic Web Services, involving the automation of service discovery, acquisition, composition and monitoring.

The relationship between web service and semantic web technology is encapsulated in Fig. 9.13 below. Combining the two technology creates the next generation of web services: semantically-enabled web services with the more sophisticated capabilities described above.

9.6.1 Web Service Modeling Ontology (WSMO)

The Web Service Modeling Ontology (WSMO) is one ontology which has been developed to provide a conceptual model for the Semantic Web Services. Another approach is OWL-S: n the interests of brevity we will focus on WSMO and refer the interested read to W3C (2004) for further information about OWL-S. In WSMO, four elements are identified as the fundamental pillars of the model: namely, *ontologies* as shared vocabularies with clearly defined semantics, *web services* as means to abstract the IT functionality provided, *goals* representing users requests referring to the problem-solving aspect of our architecture and finally *mediators* for interpretability between the various semantic descriptions.

Ontologies are used as the data model throughout WSMO, meaning that all resource descriptions as well as all data interchanged during service usage are based on ontologies. Ontologies are a widely accepted state-of-the-art knowledge representation, and have thus been identified as the central enabling technology for the Semantic Web. The extensive usage of ontologies allows semantically enabled information processing as well as support for interoperability; WSMO also supports the ontology languages defined for the Semantic Web.

Goals provide means to characterize user requests in terms of functional and non-functional requirements. For the former, a standard notion of pre and post conditions has been chosen and the later provides a predefined Ontology of generic properties. For functional aspects a standard notion of pre and post conditions has been chosen. For non-functional properties (QoS) logical expressions are used to specify the QoS values provided by the service.

Web Service descriptions specify the functionality, the means of interaction and non-functional properties (QoS) aspects provided by the Web Service. More concretely, a Web service presents:

a *capability* that is a functional description of a Web service. A set of constraints on the input and output of a service as well as constraints not directly affecting the input (preconditions) and output (postconditions) of the service but which need to hold before (assumptions) and after (effects) the execution of the service are part of the capability description.

interfaces that specify how the service behaves in order to achieve its functionality. A service interface consists of a choreography that describes the interface for the client-service interaction required for service consumption, and an orchestration that describes how the functionality of a Web service is achieved by aggregating other Web services.

non-functional properties that specify the QoS values provided by the service. Non-functional properties of services or goals are modeled in a way similar to which capabilities are currently modeled in WSMO, more precisely by means of logical expressions.

Mediators provide additional procedural elements to specify further mappings that cannot directly be captured through the usage of Ontologies. Using Ontologies provides real-world semantics to our description elements as well as machine processable formal semantics through the formal language used to specify them. The concept of Mediation in WSMO addresses the handling of heterogeneities occurring between elements that shall interoperate by resolving mismatches between different used terminologies (data level), on communicative behavior between services (protocol level), and on the business process level. A WSMO Mediator connects the WSMO elements in a loosely coupled manner, and provides mediation facilities for resolving mismatches that might arise in the process of connecting different elements defined by WSMO.

9.6.2 Web Service Modeling Language (WSML)

The Web Service Modeling Language (WSML) is a language (or more accurately a family of languages) for the description of ontologies, goals, web services and mediators, based on the conceptual model of WSMO. A major goal in the development of WSML is to investigate the applicability of different formalisms, most notably Description Logics and Logic Programming, in the area of Web services. A fuller discussion of the various WSML dialects and their advantages and disadvantages can be found in Vitvar et al. (2007).

9.6.3 Web Service Modeling Execution Environment (WSMX)

The Web Service Modeling Execution Environment (WSMX) is a reference implementation of semantic service-oriented environment that is compliant with the semantic specifications of WSMO. WSMX supports semantically enabled change functions such as dynamic discovery, selection, and mediation. WSMX also implements semantically enabled control and connection functions such as service invocation and inter-operation. WSMX is an execution environment for the dynamic discovery, selection, mediation, invocation and inter-operation of the semantic web services in a reference implementation for WSMO. The development process for WSMX includes defining its conceptual model, defining the execution semantics for the environment, describing a architecture and a software design and building a working implementation.

Figure 9.14 presents the WSMX architecture and its most important components.

In terms of functionality provided, WSMX can be seen as an aggregation of the components. The central components are the Core Component, Resource Manager, Discovery, Selection, Data and Process Mediator, Communication Manager, Choreography Engine, Web Service Modeling Toolkit, and the Reasoner.

The *Core Component* is the central component of the system connecting all the other components and managing the business logic of the system. The *Resource Manager* manages the set of repositories responsible for the persistency of all the WSMO and non-WSMO related data flowing through the system. The *Discovery* component is responsible for locating services that satisfy a specific user request. A set of discovery algorithms ranging from syntactical-based to full semantically-based matching are available in WSMX for service discovery. The *Selection* component is responsible for filtering the potential services which provide the requested functionality by considering non-functional properties values and finally selecting the best service.

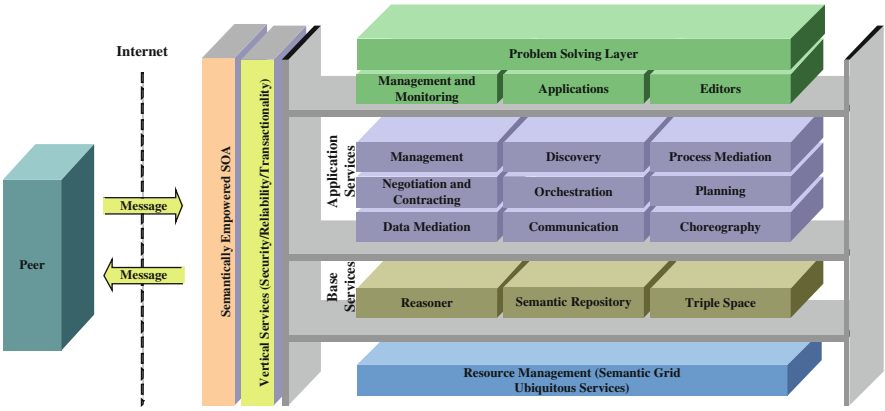

Fig. 9.14 WSMX reference architecture

WSMX provides two kinds of mediator components that deal with heterogeneity problems: a *Data Mediator* component, which mediates between different terminologies, ontologies and a *Process Mediator* component which mediates between two given patterns of interaction (i.e., WSMO choreographies) and compensates the possible mismatches that may appear. The *Communication Manager* is responsible for managing the communication between a service requester and a service provided. The two way communication is handled by the Receiver, from requester to the service, and respectively by Invoker, from the service to the requester. The *Choreography engine* provides means to store and retrieve choreography interface definitions, initiates the communication between the requester and the provider in direct correlation with the results returned by the Process Mediator, and keeps track of the communication state on both the provider and the requester sides. In addition it provides grounding information to the communication manager to enable any ordinary Web Service invocation.

The semantic inference required by each of the WSMX components is provided by the *Reasoner* component. On the client/developed side a toolkit, called the *Web Services Modeling Toolkit (WSMT)* is provided allowing modeling of WSMO elements and easy interaction with the WSMX server. WSMT contains a set of tools including the WSMO Visualizer for viewing and editing WSML documents using directed graphs, a WSMX. Management tool for managing and monitoring the WSMX environment, and a WSMX Data Mediation tool for creating mappings between WSMO ontologies are also available in WSMT.

In short, Semantic Web Services can lead to more flexible, interoperable, less costly IT systems. Space does not permit a full description of the technical details of semantic web services and the reader is referred to Chapter 10 of Davies et al. (2005) for an overview of current work in Semantic Web Services and to Vitvar et al. (2007) for a discussion of the applications of semantic technology in service-centric environments.

9.7 Ontologies and Computer Science

In the above sections we have discussed some areas of applications of lightweight ontologies in IT today. Moving beyond specific application areas and taking a broader view, the central property of semantic technology is that it offers the ability to provide machine-processable descriptions. As above, these descriptions can be of documents, fragments of documents or web services. Similarly, such descriptions could equally be descriptions of grid elements, handheld computing devices, security policies or business process elements.

In all these cases, the key points are (i) these descriptions have a well-defined meaning separate from the programs which interpret them which (ii) allows *interoperability* which would otherwise require hardwired solutions handcoded by humans.

In short, semantic technology offers the only path to web-scale interoperability; and such scalability and interoperability is surely one of the most pressing research challenges for computer science today.

9.8 Conclusion

In this chapter, we have discussed the notion of lightweight ontologies. We defined ontology and related ontology to other related knowledge organisation structures, formal and less formal. We then considered what might constitute a *lightweight* ontology. Our view of lightweight as opposed to "heavyweight" ontologies centred around the expressivity of the ontology description, rather than other possible notions such as scope, depth or computational tractability. Of course, tractability is related to expressivity in that, in general, the more expressive a formal language the less it is amenable to efficient machine processing (reasoning). Regarding scope and depth, these seem to us separate and highly context-dependent issues: a medical ontology could seem very wide and/or deep in terms of its domain coverage from the point of view of a general medical practitioner and yet much shallower from an ontological consultant or rather narrow from the point of view of a science journalist interested in medicine and many other disciplines besides, for example. Scope is orthogonal to expressivity (an ontology covering a wide domain can be more or less expressive), whereas depth is in general related in the sense that an ontology requiring more expressive language to describe it will typically be deeper (i.e. model the domain of interest to a more detailed level).

We proceeded to discuss ontologies and the semantic web, the emergence of which over the past decade has seen increased interest in ontologies and associated topics. We then looked at 3 key areas where ontologies are being used in IT systems today and briefly discussed the likely centrality of ontologies and semantic technology to computer science in the future. A number of requirements need to be fulfilled for the further uptake of ontologies:

- provision of ontologies and associated metadata: clearly a barrier to the use of ontologies is the need to manually construct an ontology for each application (or at least application domain) and then to populate the ontology with instance data). The use of knowledge discovery and human language technologies is starting to address these areas, as are the ever increasing number of domain ontologies available for re-use. As a specific example of the strides being made in this aera, Thomson-Reuters, the world's largest business information provider now offers the Open Calais[31] tool for semantically annotating large volumes of textual information, as well as providing its own information so annotated;
- production of ontology tools: there is need for mature tools supporting the ontology engineering lifecycle, preferably integrated into existing environments.

Assuming progress is made in these areas (and all are the subject of active research and development programmes), over the next decade, we can anticipate the increasing incorporation of ontology-based technology into mainstream IT systems and methods.

[31] http://www.opencalais.com/

Acknowledgments Céline van Damme is thanked for insightful discussions about folksonomies and informal ontolgoies in the enterprise. Section 9.4 is based partly on Warren and Davies (2007), while Section 9.6 is based partly on Davies et al. (2006).

References

Alonso, O. 2006. Building semantic-based applications using Oracle. Developer's Track at WWW2006, Edinburgh. http://www2006.org/programme/item.php?id=d16. Accessed May 2006.

Assem, M., V. Malaise, A. Miles, and G. Schreiber. 2006. A method to convert thesauri to SKOS. In Proceedings of the European Semantic Web Conference 2006 (ESWC 2006), Budva, Montenegro. Heidelberg: Springer.

Baader, F., D. Calvanese, D. McGuiness, D. Nardi, and P. Patel-Schneider. 2003. *The description logic handbook.* Cambridge, UK: Camridge University Press.

Beckett, D. 2004. RDF/XML syntax specification (revised). http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/.

Berners-Lee, T. 1999. *Weaving the web*. London: Orion Books.

Bontcheva, K., J. Davies, A. Duke, T. Glover, N. Kings, and I. Thurlow. 2006. Semantic information access. In *Semantic web technologies: Trends and research in ontology-based systems*, eds. J. Davies, R. Studer, and P. Warren, Chichester: John Wiley.

Brickley, D., and Guha, R.V, eds. 2000. Resource description framework (RDF) schemas, W3C. http://www.w3.org/TR/2000/CR-rdf-schema-20000327/.

Bernstein, A., E. Kaufmann, A. Goehring, and C. Kiefer. 2005. Querying ontologies: A controlled english interface for end-users, Proceedings of the 4th International Semantic Web Conference, ISWC2005, Galway, Ireland, November 2005. Heidelberg: Springer.

Borst, P., and H. Akkermans. 1997. Engineering ontologies. *International Journal of Human-Computer Studies*, 46:365–406.

Chinchor, N., and P. Robinson. 1998. MUC-7 named entity task definition (version 3.5). In Proceedings of the 7th Message Understanding Conference (MUC-7), Fairfax, VA.

Cunningham, H. 2000. Software architecture for language engineering. PhD Thesis, University of Sheffield.

Cunningham, H. 1999. Information extraction: A user guide (revised version). Department of Computer Science, University of Sheffield, May 1999.

Davies, J., D. Fensel, and F. van Harmelen. 2003. *Towards the semantic web.* Chicester: Wiley.

Davies, J., R. Weeks, and U. Krohn. 2003. QuizRDF: Search technology for the semantic web. In *Towards the semantic web*, eds. J. Davies, D. Fensel, and F. van Harmelen. Chicester: Wiley.

Davies, J., R. Studer, Y. Sure, and P. Warren. 2005. Next generation knowledge management. *BT Technology Journal* 23(3):175–190, July 2005.

Davies, J., R. Studer, and P. Warren. 2006. *Semantic web technology: Trends & research*. Chicester: Wiley.

Davies, J., et al. 2007. NESSI semantic technologies working group research roadmap 2007–2010 Version 1.2. http://www.nessi-europe.com/Nessi/LinkClick.aspx?fileticket=Ja5zTnzK4%2fM%3d&tabid=241&mid=694. Accessed 10th June 2007.

DCMI Usage Board. 2005. DCMI metadata terms. http://dublincore.org/documents/2005/06/13/dcmi-terms/.

Dean, M., and G. Schreiber, eds.; Bechhofer, S., F. van Harmelen, J. Hendler, I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider, and L.A. Stein, 2004. OWL web ontology language reference. W3C recommendation. http://www.w3.org/TR/owl-ref/. Accessed 10 February 2004.

Domingue, J., M. Dzbor, and E. Motta. 2004. Collaborative semantic web browsing with magpie. In *The semantic web: Research and applications*, Proceedings of ESWS, 2004, LNCS 3053, eds. J. Davies, C. Bussler, D. Fensel, and R. Studer, 388–401. Heidelberg: Springer.

Dumais, S., E. Cutrell, J. Cadiz, G. Jancke, R. Sarin, and D. Robbins. 2003. Stuff I've Seen: A system for personal information retrieval and re-use. In Proceedings of SIGIR'03, Toronto. New York, NY: ACM Press.

Dill, S., N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, K.S. McCurley, S. Rajagopalan, A. Tomkins, J.A. Tomlin, and J.Y. Zienberer. 2003. A case for automated large scale semantic annotation. *Journal of Web Semantics* 1(1):115–132.

Ding, L., T. Finin, A. Joshi, R. Pan, R.S. Cost, Y. Peng, P. Reddivari, V. Doshi, and J. Sachs. 2004. Swoogle: A search and metadata engine for the semantic web, Conference on Information and Knowledge Management CIKM04, Washington, DC, November 2004.

Ehrig, M., P. Haase, M. Hefke, and N. Stojanovic. 2005. Similarity for ontologies – A comprehensive framework. In Proceedings of the 13th European Conference on Information Systems, Regensburg, May 2005.

Fensel, D., and F. van Harmelen. 2007. Unifying reasoning and search to web scale. *IEEE Internet Computing* 11(2):94–96.

Fensel, D., and M. Musen. 2001. The semantic web: A brain for humankind. *IEEE Intelligent Systems* 16(2):24–25.

Giunchiglia, F., M. Marchese, and I. Zaihrayeu. 2005. Encoding classification into lightweight ontologies. In Proceedings of the 2nd European Semantic Web Conference ESWC05, Heraklion, Crete, May 2005.

Glaser, H., H. Alani, L. Carr, S. Chapman, F. Ciravegna, A. Dingli, N. Gibbins, S. Harris, M.C. Schraefel, and N. Shadbolt. 2004. CS AKTiveSpace: Building a semantic web application. In *The semantic web: Research and applications*, Proceedings of ESWS, 2004, LNCS 3053, eds. J. Davies, C. Bussler, D. Fensel, and R. Studer, 388–401. Heidelberg: Springer.

Grishman, R. 1997. TIPSTER architecture design document version 2.3. Technical report, DARPA. http://www.itl.nist.gov/iaui/894.02/related_projects/tipster/.

Gruber, T.R. 1992. A translation approach to portable ontologies. *Knowledge Acquisition* 5(2):199–220. http://ksl-web.stanford.edu/KSL_Abstracts/KSL-92-71.html. Accessed 1993.

Gruber, T.R. 1993. Toward principles for the design of ontologies used for knowledge sharing. In International Workshop on Formal Ontology, Padova, Italy, eds. N. Guarino and R. Poli. http://ksl-web.stanford.edu/KSL_Abstracts/KSL-93-04.html.

Guarino, N., and P. Giaretta. 1995. Ontologies and knowledge bases: Towards a terminological clarification. In *Towards very large knowledge bases: Knowledge building and knowledge sharing*, ed. N. Mars, 25–32. Amsterdam: IOS Press.

Guarino, N. 1998. Formal ontology in information systems. In Proceedings of FOIS'98, Trento, Italy, 6–8 June 1998, ed. N. Guarino, 3–15. Amsterdam: IOS Press.

Guha, R., and R. McCool. 2003. Tap: A semantic web platform. *Computer Networks* 42:557–577.

Guha, R., R. McCool, and E. Miller. 2003. Semantic search. In WWW2003, Budapest, Hungary, 20–24 May 2003.

Iosif, V., P. Mika, R. Larsson, and H. Akkermans. 2003. Field experimenting with semantic web tools in a virtual organisation. In *Towards the semantic web*, eds. J. Davies, D. Fensel, and F. van Harmelen. Chicester: Wiley.

Jansen, B.J., A. Spink, and T. Saracevic. 2000. Real life, real users, and real needs: A study and analysis of user queries on the web. *Information Processing and Management* 36(2): 207–227.

Kiryakov, A. 2006. Ontologies for knowledge management. In *Semantic web technologies: Trends and research in ontology-based systems*, Chapter 7, eds. J. Davies, R. Studer, and P. Warren. Chicester: Wiley.

Kiryakov A., and K.Iv. Simov. 1999. Ontologically supported semantic matching. In Proceedings of the "NODALIDA'99: Nordic Conference on Computational Linguistics", Trondheim, 9–10 Dec 1999.

Kiryakov, A., B. Popov, D. Ognyanov, D. Manov, A. Kirilov, and M. Goranov. 2004. Semantic annotation, indexing, and retrieval. *Elsevier's Journal of Web Semantics* 1, ISWC2003 special issue (2). http://www.websemanticsjournal.org/.

Kiryakov, A., D. Ognyanov, and D. Manov. 2005. OWLIM – A pragmatic semantic repository for OWL. In Proceedings of International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2005), WISE, 20 Nov 2005, New York City.

Klyne, G., and J.J. Carroll. 2004. Resource description framework (RDF): Concepts and abstract syntax. W3C recommendation. http://www.w3.org/TR/rdf-concepts/. Accessed 10 Feb 2004.

Landauer T., and S. Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review* 104(2):211–240.

McCarthy, J. 1980. Circumscription – A form of non-monotonic reasoning. *Artificial Intelligence* 13:27–39.

Mahesh, K., J. Kud, and P. Dixon. 1999. Oracle at TREC8: A lexical approach. In Proceedings of the 8th Text Retrieval Conference (TREC-8) Gaithersburg, Maryland.

Mangold, C. 2007. A survey and classification of semantic search approaches. *International Journal of Metadata, Semantics and Ontologies* 2(1):23–34.

McDermott, D. (1978). Tarskian semantics, or no notation without denotation! *Cognitive Science* 2:277–282.

Miles, A., and D. Brickley, eds. 2005. SKOS core guide. WorldWideWeb consortium. Latest version. http://www.w3.org/TR/swbp-skos-core-guide.

Pollock, J., and R. Hodgson. 2004. *Adaptive information: Improving business through semantic interoperability, grid computing, and enterprise integration*. Wiley-Interscience.

Popov, B., A. Kiryakov, A. Kirilov, D. Manov, D. Ognyanoff, and M. Goranov. 2003. KIM – Semantic annotation platform. In Proceedings of 2nd International Semantic Web Conference (ISWC2003), Florida, 20–23 Oct 2003, LNAI Vol. 2870, 834–849. Berlin/Heidelberg: Springer.

Rocha, C., D. Schwabe, M.P., and de Aragao. 2004. A hybrid approach for searching in the semantic web. WWW 2004, New York, 17–22 May 2004.

Salton, G., A. Wong, and C.S. Yang. 1997. A vector space model for automatic indexing. In *Readings in information retrieval*, eds. K. Sparck-Jones, and P. Willett. San Fransisco, CA: Morgan-Kaufman.

Smith, B. 2003. Ontology. In *Blackwell guide to the philosophy of computing and information*, ed. L. Floridi, 155–166. Oxford: Blackwell.

Smith, B., and C. Welty. 2001. Ontology: Towards a new synthesis. In Proceedings of the FOIS'01, Ogunquit, ME.

Sparck-Jones, K., and P. Willett. 1997. Readings in information retrieval. San Fransisco, CA: Morgan-Kaufman.

Spark Jones, K. 2004. What's new about the semantic web? Some questions. *SIGIR Forum* 38(2). http://www.sigir.org/forum/2004D-TOC.html. Accessed Dec 2004.

Specia, L., and E. Motta. 2007. *Integrating folksonomies with the semantic web*. In Proceedings of the European Semantic Web Conference 2007 (ESWC 2007), Innsbruck, Austria. Heidelberg: Springer.

Terziev, I., A. Kiryakov, and D. Manov. 2004. D1.8.1. Base upper-level ontology (BULO) guidance, report EU-IST integrated project (IP) IST-2003-506826 (SEKT). http://proton.semanticweb.org/D1_8_1.pdf.

Vallet, D., M. Fernandez, and P. Castells. 2005. An ontology-based information retrieval model. In Proceedings of the 2nd European Semantic Web Conference, ESWC2005, Heraklion, Crete, May/June 2005, LNCS 3532/2005, eds. A. Gómez-Pérez, and J. Euzenat. Berlin: Springer.

van Damme, C., M. Hepp, and K. Siorpaes. 2007. FolksOntology: An integrated approach for turning folksonomies into ontologies. Bridging the Gap Between Semantic Web and Web 2.0 Workshop, 4th European Semantic Web Conference, Innsbruck, Austria, June 2007.

van Damme, C., T. Cornen, and E. Vandijck. 2008. Turning a corporate folksonomy into a lightweight corporate ontology. 11th International Business Information Systems Conference, BIS 2008, Innsbruck, Austria, May 2008. Heidelberg: Springer.

van Ossenbruggen, J., L. Hardman, and L. Rutledge. 2002. Hypermedia and the semantic web: A research agenda. *Journal of Digital Information* 3(1), May 2002.

Vitvar, T., M. Zaremba, M. Moran, and D. Fensel. 2007. SESA: Emerging technology for service-centric environments. *IEEE Software* 24(6):56–67, Nov/Dec 2007.

Voorhees, E. 1998. Using WordNet for text retrieval. In *WordNet: An electronic lexical database*, ed. C. Fellbaum. Cambridge, MA: MIT Press.

Warren, P., and J. Davies. 2007. Managing the risks from information through semantic information management. *BT Technology Journal* 25(1):178–191, Jan 2007.

W3C Member Submission. 2004. OWL-S: Semantic markup for web services. http://www.w3.org/Submission/OWL-S/.