Semantic Web

Ian Horrocks, Sean Bechhofer

University of Manchester
Oxford Road
Manchester M13 9PL, UK
horrocks@cs.man.ac.uk, sean.bechhofer@manchester.ac.uk

Summary. The Semantic Web aims to explicate the meaning of web content by adding semantic annotations that describe the content and function of resources. Providing shareable annotations requires the use of ontologies that describe a common model of a domain. The Web Ontology Language OWL has been defined in order to support representation of ontologies, and their manipulation through the use of reasoning. We provide a brief overview of OWL and the underlying theory, describe applications of ontologies, and give pointers to areas of overlap between Semantic Web and Accessibility research.

1 Introduction

While phenomenally successful in terms of size and number of users, today's World Wide Web is fundamentally a relatively simple artefact. Web content consists mainly of distributed hypertext, and is accessed via a combination of keyword based search and link navigation. This simplicity has been one of the great strengths of the Web, and has been an important factor in its popularity and growth: naive users are able to use it, and can even create their own content.

The explosion in both the range and quantity of Web content has, however, highlighted some serious shortcomings in the hypertext paradigm. In the first place, the required content becomes increasingly difficult to locate using the search and browse paradigm. Finding information about people with very common names (or with famous namesakes) can, for example, be a frustrating experience. More complex queries can be even more problematical: a query for "animals that use sonar but are neither bats nor dolphins" may either return many irrelevant results related to bats and dolphins (because the search engine failed to understand the negation), or may fail to return many relevant results (because most relevant Web pages *also* mention bats or dolphins). More complex tasks may be extremely difficult, or even impossible. Examples of such tasks include locating information in data repositories that are not directly accessible to search engines (Volz et al. 2004), or finding and using so-called web services (McIlraith et al. 2001).

If human users have difficulty accessing web content, the problem is even more severe for automated processes. This is because web content is primarily intended

for presentation to and consumption by human users: HTML markup is mainly concerned with layout, size, colour and other presentational issues. Moreover, web pages increasingly use images, often including active links, to present information. Human users are able to interpret the significance of such features, and thus understand the information being presented, but this may not be so easy for an automated process or "software agent".

The Semantic Web aims to overcome some of the above mentioned problems by making web content more accessible to automated processes; the ultimate goal is to transform the existing web into "... a set of connected applications ... forming a consistent logical web of data ..." (Berners-Lee 1998). This is to be achieved by adding *semantic annotations* to Web content, i.e., annotations that describe the meaning of the content.

In the remainder of this chapter we will examine in a little more detail what semantic annotations will look like, how they describe meaning, and how automated processes can exploit such descriptions. We will also discuss the impact of the Semantic Web and Semantic Web technology on accessibility.

2 Background

As we mentioned above, the key idea behind the semantic web is to explicate the *meaning* of web content by adding semantic annotations. If we assume for the sake of simplicity that such annotations take the form of XML style tags, we could imagine a fragment of a web page being annotated as follows:

⟨`Wizard`⟩Harry Potter⟨`/Wizard`⟩ has a pet called
⟨`SnowyOwl`⟩Hedwig⟨`/SnowyOwl`⟩.

Taken in isolation, however, such annotations are of only limited value: the problem of understanding the terms used in the text has simply been transformed into the problem of understanding the terms used in the labels. A query for information about raptors, for example, may not retrieve this text, even though owls are raptors. This is where *ontologies* come into play: they provide a mechanism for introducing a vocabulary and giving precise meanings to the terms in the vocabulary. A suitable ontology might, for example, introduce the term SnowyOwl, and include the information that a SnowyOwl is a kind of Owl, and that an Owl is a kind of Raptor. Moreover, if this information is represented in a way that is accessible to our query engine, then it would be able to recognise that the above text is relevant to our query about raptors.

Ontology, in its original pholosophical sense, is a fundamental branch of metaphysics focussing on the study of existence; its objective is to determine what entities and types of entities actually exist, and thus to study the structure of the world. The study of ontology can be traced back to the work of Plato and Aristotle, and from the very beginning included the development of hierarchical categorisations of different kinds of entity and the features that distinguish them: the well known "tree of Porphyry", for example, identifies animals and plants as sub-categories of living things distinguished by animals being *sensitive*, and plants being *insensitive* (see Figure 1).

In computer science, an ontology is usually taken to be a model of (some aspect of) the world; it introduces vocabulary describing various aspects of the domain

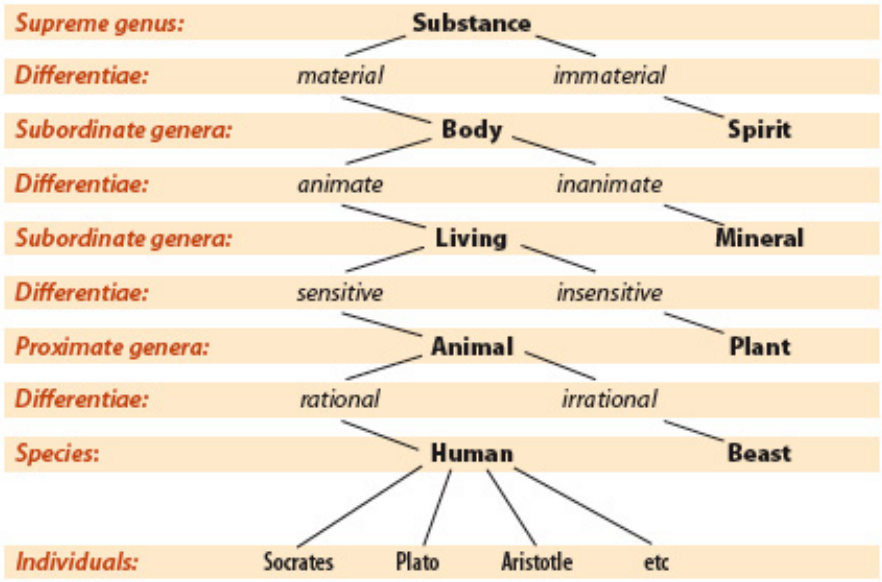

Fig. 1. Tree of Porphyry.

being modelled, and provides an explicit specification of the intended meaning of the vocabulary. This specification often includes classification based information not unlike that in Porphyry's famous tree. For example, Figure 2 shows a screenshot of a Pizza ontology as displayed by the Protégé ontology design tool (Knublauch et al. 2004). The ontology introduces various pizza related vocabulary (some of which can be seen in the left hand panel), such as "NamedPizza" and "RealItalianPizza", and arranges it hierarchically: RealItalianPizza is, for example, a sub-category of NamedPizza. The other panels display information about the currently selected category, RealItalianPizza in this case, describing its meaning: a RealItalianPizza is a Pizza whose country of origin is Italy; moreover, a RealItalianPizza always has a ThinAndCrispyBase. Ontologies can be used to annotate and to organise data from the domain: if our data includes instances of RealItalianPizza, then we can return them in response to a query for instances of NamedPizza.

3 The Web Ontology Language OWL

The architecture of the Web depends on agreed standards such as HTTP that allow information to be shared and exchanged. A standard ontology language is, therefore, a prerequisite if ontologies are to be used in order to share and exchange *meaning*. Recognising this fact, the World Wide Web Consortium (W3C) set up a standardisation working group to develop such a langauge. The result of this activity was the Web Ontology Language OWL ontology language standard (Patel-Schneider et al. 2004). OWL exploited existing work on langauges such as OIL (Fensel et al. 2001) and DAML+OIL (Horrocks et al. 2002) and, like them, was based on a Descrip-

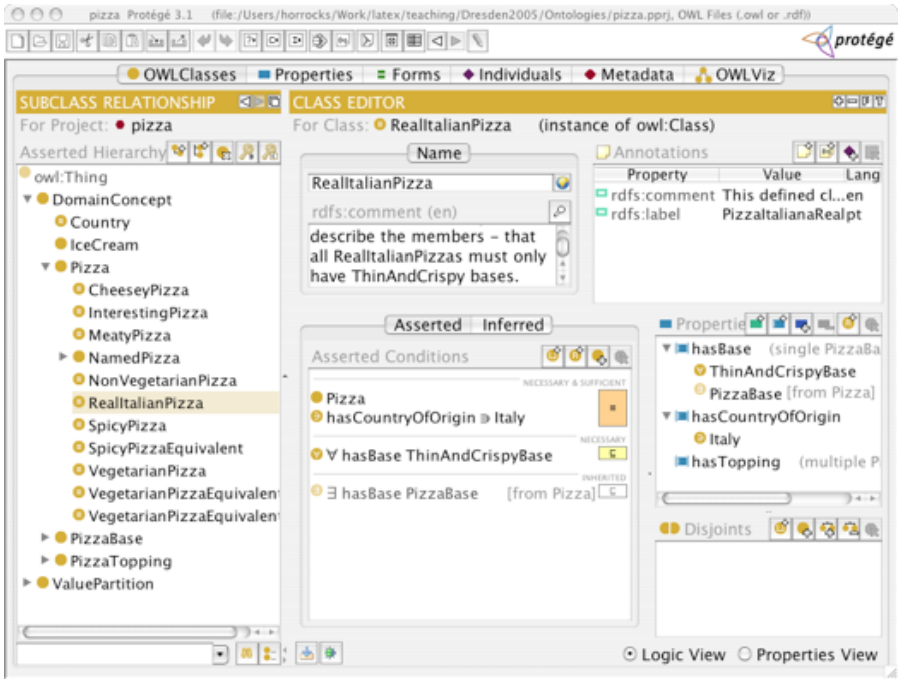

Fig. 2. Example pizza ontology.

tion Logic (DL). In the following we will briefly introduce DLs and OWL. For more complete information the reader should consult The Description Logic Handbook (Baader et al. 2003), and the OWL specification (Patel-Schneider et al. 2004).

3.1 Description Logic

Description logics (DLs) are a family of logic-based knowledge representation formalisms; they are descendants of Semantic Networks (Woods 1985) and KL-ONE (Brachman and Schmolze 1985). These formalisms all adopt an object-oriented model, similar to the one used by Plato and Aristotle, in which the domain is described in terms of individuals, *concepts* (usually called *classes* in ontology languages), and *roles* (usually called relationships or properties in ontology languages). Individuals, e.g., "Socrates" are the basic elements of the domain; concepts, e.g., "Human", describe sets of individuals having similar characteristics; and roles, e.g., "hasPupil" describe relationships between pairs of individuals, such as "Socrates hasPupil Plato".

As well as *atomic* concept names such as Human, DLs also allow for concept descriptions to be composed from atomic concepts and roles. Moreover, it is possible to assert that one concept (or concept description) is subsumed by (is a sub-concept of), or is exactly equivalent to, another. This allows for easy extension of the vocabulary by introducing new names as abbreviations for descriptions. For example, using standard DL notation, we might write:

$$\mathsf{HappyParent} \equiv \mathsf{Parent} \sqcap \forall \mathsf{hasChild}.(\mathsf{Intelligent} \sqcup \mathsf{Athletic})$$

This introduces the concept name HappyParent, and asserts that its instances are just those individuals that are instances of Parent, and all of whose children are instances of either intelligent or athletic.

Another distinguishing feature of DLs is that they are logics, and so have a formal semantics. DLs can, in fact, be seen as decidable subsets of first-order predicate logic, with individuals being equivalent to constants, concepts to unary predicates and roles to binary predicates. As well as giving a precise and unambiguous meaning to descriptions of the domain, this also allows for the development of reasoning algorithms that can be used to answer complex questions about the domain. An important aspect of DL research has been the design of such algorithms, and their implementation in (highly optimised) reasoning systems that can be used by applications to help them "understand" the knowledge captured in a DL based ontology. We will return to this point in Section 4.

A given DL is characterised by the set of constructors provided for building concept descriptions. These typically include at least intersection ($\sqcap$), union ($\sqcup$) and complement ($\neg$), as well as restricted forms of existential ($\exists$) and universal ($\forall$) quantification, which in OWL are called, respectively, *someValuesFrom* and *allValuesFrom* restrictions. OWL is based on a very expressive DL called $\mathcal{SHOIN}$ that also provides cardinality restrictions ($\geqslant$, $\leqslant$) and enumerated classes (called *oneOf* in OWL) (Horrocks et al. 2003, Horrocks and Sattler 2005). Cardinality restrictions allow, e.g., for the description of a concept such as people who have at least two children, while enumerated classes allow for classes to be described by simply enumerating their instances, e.g.,:

$$\mathsf{EUcountries} \equiv \{\mathsf{Austria}, \ldots, \mathsf{UK}\}$$

$\mathcal{SHOIN}$ also provides for transitive roles, allowing us to state, e.g., that if x has an ancestor y and y had an ancestor z, then z is also an ancestor of x, and for inverse roles, allowing us to state, e.g., that if z is an ancestor of x, then x is also an descendent of z. The constructors provided by OWL, and the equivalent DL syntax, are summarised in Figure 3.

Constructor	DL Syntax	Example
`intersectionOf`	$C_1 \sqcap \ldots \sqcap C_n$	$\mathsf{Human} \sqcap \mathsf{Male}$
`unionOf`	$C_1 \sqcup \ldots \sqcup C_n$	$\mathsf{Doctor} \sqcup \mathsf{Lawyer}$
`complementOf`	$\neg C$	$\neg\mathsf{Male}$
`oneOf`	$\{x_1 \ldots x_n\}$	$\{\mathsf{john}, \mathsf{mary}\}$
`allValuesFrom`	$\forall P.C$	$\forall\mathsf{hasChild}.\mathsf{Doctor}$
`someValuesFrom`	$\exists r.C$	$\exists\mathsf{hasChild}.\mathsf{Lawyer}$
`hasValue`	$\exists r.\{x\}$	$\exists\mathsf{citizenOf}.\{\mathsf{USA}\}$
`minCardinality`	$(\geqslant n\ r)$	$(\geqslant 2\ \mathsf{hasChild})$
`maxCardinality`	$(\leqslant n\ r)$	$(\leqslant 1\ \mathsf{hasChild})$
`inverseOf`	r^-	$\mathsf{hasChild}^-$

Fig. 3. OWL constructors

In DLs it is usual to separate the set of statements that establish the vocabulary to be used in describing the domain (what we might think of as the schema) from the set of statements that describe some particular situation that instantiates the schema (what we might think of as data); the former is called the TBox (Terminology Box), and the latter the ABox (Assertion Box). An OWL ontology is simply equivalent to a set of $\mathcal{SHOIN}$ TBox and ABox statements. This mixing of schema and data is quite unusual (in fact ontologies are usually thought of as consisting only of the schema part), but does not affect the meaning—from a logical perspective, $\mathcal{SHOIN}$ KBs and OWL ontologies are just sets of axioms.

The main difference between OWL and $\mathcal{SHOIN}$ is that OWL ontologies use an RDF based syntax intended to facilitate its use in the context of the Semantic Web. This syntax is rather verbose, and not well suited for presentation to human beings. E.g., the description of HappyParent given above would be written in OWL's RDF syntax as follows:

```
<owl:Class>
  <owl:intersectionOf rdf:parseType=" collection">
    <owl:Class rdf:about="#Parent"/>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasChild"/>
      <owl:allValuesFrom>
        <owl:unionOf rdf:parseType=" collection">
          <owl:Class rdf:about="#Intelligent"/>
          <owl:Class rdf:about="#Athletic"/>
        </owl:unionOf>
      </owl:allValuesFrom>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

4 Ontology Reasoning

We mentioned in Section 3.1 that the design and implementation of reasoning systems is an important aspect of DL research. The availability of such reasoning systems was one of the motivations for basing OWL on a DL. This is because reasoning is essential in supporting both the design of high quality ontologies, and the deployment of ontologies in applications.

4.1 Reasoning at design time

Ontologies may be very large and complex: the well known Snomed clinical terms ontology includes, for example, more than 200,000 class names (Spackman 2000). Building and maintaining such ontologies is very costly and time consuming, and providing tools and services to support this "ontology engineering" process is of crucial importance to both the cost and the quality of the resulting ontology. State of the art ontology development tools, such as SWOOP (Kalyanpur et al. 2005a) and Protégé (Knublauch et al. 2004), therefore use a DL reasoner, such as FaCT++ (Tsarkov and Horrocks 2006), Racer (Haarslev and Möller 2001) or Pellet (Sirin et al.

2005), to provide feedback to the user about the logical implications of their design. This typically includes (at least) warnings about inconsistencies and redundancies.

An inconsistent (sometimes called unsatisfiable) class is one whose description is "over-constrained", with the result that it can never have any instances. This is typically an unintended feature of the design—why introduce a name for a class that can never have any instances—and may be due to subtle interactions between descriptions. The ability to detect such classes and bring them to the attention of the ontology engineer is, therefore, a very useful feature.

It is also possible that the descriptions in the ontology mean that two classes necessarily have exactly the same set of instances, i.e., that they are alternative names for the same class. This may be desirable in some situations, e.g., to capture the fact that "Myocardial infarction" and "Heart attack" mean the same thing. It could, however, also be the inadvertent result of interactions between descriptions, and so it is also useful to be able to alert users to the presence of such "synonyms".

In addition to checking for inconsistencies and synonyms, ontology development tools usually also check for implicit subsumption relationships, and amend the class hierarchy accordingly. This is also a very useful design aid: it allows the ontology developer to focus on class descriptions, leaving the computation of the class hierarchy to the reasoner, and it can also be used by the developer to check if the hierarchy induced by the class descriptions is consistent with their intuition.

Recent work has also shown how reasoning can be used to support modular design (Cuenca Grau et al. 2007b) and module extraction (Cuenca Grau et al. 2007a), important techniques for working with large ontologies. When developing a large ontology such as SNOMED, it is useful if not essential to divide the ontology into modules, e.g., to facilitate parallel work by a team of ontology developers. Reasoning techniques can be used to alert the developers to unanticipated and/or undesirable interactions between the various modules. Similarly, it may be desirable to extract from a large ontology a smaller module containing all the information relevant to some subset of the domain, e.g., heart disease—the resulting small(er) ontology will be easier for humans to understand and easier for applications to use. Reasoning can be used to compute a module that is as small as possible while still containing all the necessary information.

Finally, in order to maximise the benefit of all these services, a modern system should also be able to explain its inferences: without this facility, users may find it difficult to repair errors in the ontology and may even start to doubt the correctness of the reasoning system. Explanation typically involves computing a (hopefully small) subset of the ontology that still entails the inference in question, and if necessary presenting the user with a chain of reasoning steps (Kalyanpur et al. 2005b).

4.2 Reasoning in deployment

Reasoning is also important when ontologies are deployed in applications—it is needed, e.g., in order to answer structural queries about the domain and to retrieve data. If we assume, for example, an ontology that includes the above description of HappyParent, and we know that John is a HappyParent, that John has a child Mary (i.e., John hasChild Mary), and that Mary is not Athletic, then we would like to be able to infer that Mary is Intelligent.

The above example may seem quite trivial, but it is easy to imagine that, with large ontologies, query answering may be a very complex task. The use of DL rea-

soners allows OWL ontology applications to answer complex queries, and to provide guarantees about the correctness of the result. This is particularly important if ontology based systems are to be used as components in larger applications, such as the Semantic Web, where the correct functioning of automated processes may depend on their being able to (correctly) answer such queries.

5 Ontology Applications

The availability of tools and reasoning systems such as those mentioned in Section 4 has contributed to the increasingly widespread use of OWL, not only in the Semantic Web per se, but as a popular language for ontology development in fields as diverse as biology (Sidhu et al. 2005), medicine (Golbreich et al. 2006), geography (Goodwin 2005), geology (SWEET), astronomy (Derriere et al. 2006), agriculture (Soergel et al. 2004) and defence (Lacy et al. 2005). Applications of OWL are particularly prevalent in the life sciences where it has been used by the developers of several large biomedical ontologies, including the Biological Pathways Exchange (BioPAX) ontology (Ruttenberg et al. 2005), the GALEN ontology (Rector and Rogers 2006), the Foundational Model of Anatomy (FMA) (Golbreich et al. 2006), and the National Cancer Institute thesaurus (Hartel et al. 2005).

The importance of reasoning support in such applications was highlighted in (Kershenbaum et al. 2006), which describes a project in which the Medical Entities Dictionary (MED), a large ontology (100,210 classes and 261 properties) that is used at the Columbia Presbyterian Medical Center, was converted into OWL, and checked using an OWL reasoner. This check revealed "systematic modelling errors", and a significant number of missed subClass relationships which, if not corrected, "could have cost the hospital many missing results in various decision support and infection control systems that routinely use MED to screen patients".

6 Semantics and Accessibility

The development of Semantic Web languages and technology was primarily driven by a desire to overcome the problems encountered by "software agents" in using information available on the web. However, the chief characteristic of the Semantic Web approach, namely the annotation of resources with machine readable descriptions also offers a promise for accessibility. Principled separation of content from presentational information (e.g. through the use of CSS) helps to alleviate some problems – for example ensuring that presentational aspects are not used to convey additional meaning. Rich, semantic annotations of content can push this further – explicitly publishing content in machine readable forms opens up the possibilities for end user applications to transform the annotated information and present it to a user in an appropriate fashion.

We can identify at least three areas where accessibility issues relate to Semantic Web. First of all, Semantic Web end-user applications targeted at *consumers* must be sympathetic to the needs of users. There are also an increasing number of tools aimed at *producers* of Semantic information – again, we must be careful in the design and execution of these applications. Finally, there is the possibility of using Semantic

Web technologies and approaches in supporting access to content. We briefly discuss each of these issues.

6.1 End User Applications

A number of applications (for example Magpie (Dzbor et al. 2003) or COHSE (Carr et al. 2001, Yesilada et al. 2006)) provide what we might call Semantic Web Browsers. These provide enhanced navigational possibilities for users, based on additional semantic information which is either embedded in pages, added through annotations, or gleaned at run time through the use of natural language processing techniques. While browsing a web resource, the applications give additional context or links to related resources. These applications tend to use client side processing (for example relying on dynamic HTML or AJAX-style interactions) in order to provide an enhanced user experience. Clearly this raises questions as to the accessibility of the presentations generated. Similarly, applications and tools that support browsing of RDF repositories tend to be graphical in nature. To date, the issue of accessibility has not been explicitly tackled within such applications.

6.2 Support Tools

Tools are also available to support producers of semantic information. As discussed in Section 3.1, the normative presentation syntax for OWL (XML/RDF) is a rather verbose format which is not particularly human readable. Tooling is thus required to support editing and manipulation of ontologies. In particular, ontology editors such as Protégé and SWOOP allow the development, construction and maintenance of ontologies. However, these tools are largely *graphical* in nature, and present potential difficulties for visually impaired users. As with end-user tools, little exploration of accessible ontology development interfaces has been done to date – ontology editors are still perhaps something of a niche-market. Most modern ontology development tools have been developed using Java, so the possibility exists for enhancement of the interfaces (for example through the Java Accessibility API), but additional care is likely to be required in the interface design.

6.3 Annotation for Accessibility

Finally, we turn our attention to the use of Semantic Web technology to support better access to information. The Semantic Web is built on the notation of annotation or decoration of resources with additional information describing the content or function of those resources. This explicit representations of information allows applications or software agents to perform actions on behalf of users.

Improving sharing and interoperability between applications is seen as a key benefit of the use of ontologies. EARL (Abou-Zahra 2005) uses vocabularies in order to facilitate the exchange of information between tools.

Although most examples of semantic web applications focus on tasks such as searching or information integration, semantic annotation has been applied in order to support Web content transcoding. Annotations for Web content transcoding aim to provide better support either for audio rendering, and thus for visually impaired users, or for visual rendering in small screen devices.

Proxy-based systems to transcode Web pages based on external annotations for visually impaired users have been proposed (Takagi and Asakawa 2000, Asakawa and Takagi 2000). The main focus is on extracting visually fragmented groupings, their roles and importance. There is no particular attempt to provide a deep understanding or analysis of the page. Approaches such as SWAP (Seeman 2004) use semantic annotations to support accessibility and device independence. The SeE-Browser (Kouroupetroglou et al. 2006) consumes annotations in order to support visually impaired users' navigation around pages. Interestingly, the ontology editor reported in (Kouroupetroglou et al. 2006) is very much a visual tool (as discussed above).

DANTE (Yesilada et al. 2004) used an ontology known as WAfA to provide terms relating to mobility of visually impaired users. Annotations made on pages describe the roles that particular elements may play. These annotations can then drive a transformation process. The use of an ontology helps to guarantee a consistency across annotations and their interpretation. DANTE relies on the annotation of individual pages, however, which can be costly.

An alternative approach is adopted by SADIe (Harper and Bechhofer 2005, Bechhofer et al. 2006), which relies on annotations of style sheet information attached to pages. The rationale behind SADIe's approach is that the classes that appear in Cascading Style Sheet definitions often have some implicit semantics associated with them. For example, a CSS class *menu* may be used to define the presentational attributes associated with a menu appearing on a page. Such an object is likely to be important in supporting navigation around a site, and so should be given a prominent rendering in a transcoded page. The ontology provides an abstraction over the roles of elements appearing in pages, and allows applications to apply general transformations over different sites. The structure of the ontology also allows the description of general rules – e.g. menu items should be promoted to a prominent position – along with specialisations of those items – e.g. items in a navigation bar are menu items.

These annotations differ slightly from mainstream Semantic Web annotation approaches (Handschuh and Staab 2003) which tend to focus on annotation of content rather than structure. This is still, however, an example of the explicit exposure of information in machine readable forms. The approach of treating annotations as first class citizens, separate from the resources they annotate, is of benefit here, however, allowing third parties potential opportunities to improve access to resources where the original provider will not, or can not alter existing content.

7 Future Directions

As we have seen in Section 5, OWL is already being successfully used in many applications. This success brings with it, however, many challenges for the future development of both the OWL language and OWL tool support. Central to these is the familiar tension between the requirements for advanced features, in particular increased expressive power, and raw performance, in particular the ability to deal with very large ontologies and data sets.

Use of OWL in the life sciences domain has brought to the fore examples of both of the above mentioned requirements. On the one hand, ontologies describing complex systems in medicine and biology often require expressive power beyond what

is currently supported in OWL. Two particular features that are very often requested are the ability to "qualify" cardinality constraints, e.g., to describe the hand as having four parts *that are fingers* and one part *that is a thumb*, and the ability to have some characteristics be transferred across transitive part-whole relations, e.g., to capture the fact that a disease affecting a part of an organ affects the organ as a whole. The former feature (so called qualified cardinality restrictions) has long been well understood, and has been available for some time in DL reasoners; the latter feature is now also well understood, thanks to recent theoretical work in the DL community (Horrocks and Sattler 2004, Horrocks et al. 2006), and has recently been implemented in DL reasoners.

This happy coincidence of user requirements and extensions in the underlying DLs and reasoning systems has led to a proposal to extend OWL with these and other useful features that have been requested by users, for which effective reasoning algorithms are now available, and that OWL tool developers are willing to support. In addition to those mentioned above, the new features include extra syntactic sugar, extended datatype support, simple metamodelling, and extended annotations. The extended language, called OWL 1.1, is now a W3C member submission[1], and is already supported by tools such as Swoop, Protégé and TopBraid Composer.

As well as increased expressive power, applications may also bring with them requirements for scalability that are a challenge to current systems. This may include the ability to reason with very large ontologies, perhaps containing 10s or even 100s of thousands of classes, and the ability to use an ontology with very large data sets, perhaps containing 10s or even 100s of millions of individuals—in fact data sets much larger than this will certainly be a requirement in some applications. Researchers are rising to these challenges by developing new reasoning systems such as the OWL Instance Store (Bechhofer et al. 2005), that uses a combination of DL reasoning and relational database systems to deal with large volumes of instance data, HermiT[2], that uses a hypertableau based technique to deal more effectively with large and complex ontologies, and Kaon2 (Hustadt et al. 2004), that reduces OWL ontologies to disjunctive datalog programs, and uses deductive database techniques to enable it to deal with very large data sets.

8 Conclusions

As we have seen, the goal of Semantic Web research is to transform the Web from a linked document repository into a distributed knowledge base and application platform, thus allowing the vast range of available information and services to be more effectively exploited. As a first step in this transformation, languages such as OWL have been developed; these languages are designed to capture the knowledge that will enable applications to better understand Web accessible resources, and to use them more intelligently. As we have seen in Section 6, the annotation of resources with machine readable descriptions also offers a promise for accessibility.

Although fully realising the Semantic Web still seems some way off, OWL has already been very successful, and has rapidly become a de facto standard for ontology development in fields as diverse as geography, geology, astronomy, agriculture,

[1] `http://www.w3.org/Submission/2006/10/`
[2] `http://www.cs.man.ac.uk/~bmotik/HermiT/`

defence and the life sciences. An important factor in this success has been the availability of sophisticated tools with built in reasoning support.

The use of OWL in large scale applications has brought with it new challenges, both with respect to expressive power and scalability, but recent research has also shown how the OWL language and OWL tools can be extended and adapted to meet these challenges.

References

Shadi Abou-Zahra. Semanticweb enabled web accessibility evaluation tools. In *W4A '05: Proceedings of the 2005 International Cross-Disciplinary Workshop on Web Accessibility (W4A)*, pages 99–101, New York, NY, USA, 2005. ACM Press. ISBN 1-59593-219-4. doi: http://doi.acm.org/10.1145/1061811.1061830.

Chieko Asakawa and Hironobu Takagi. Annotation-based transcoding for nonvisual web access. In *Proceedings of the Fourth International ACM Conference on Assistive Technologies*, pages 172–179. ACM Press, 2000.

Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2003.

Sean Bechhofer, Ian Horrocks, and Daniele Turi. The OWL instance store: System description. In *Proc. of the 20th Int. Conf. on Automated Deduction (CADE-20)*, Lecture Notes in Artificial Intelligence, pages 177–181. Springer, 2005.

Sean Bechhofer, Simon Harper, and Darren Lunn. SADIe: Semantic Annotation for Accessibility. In I. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, editors, *Proceedings of ISWC2006, the Fifth International Semantic Web Conference*, volume 4273 of *Lecture Notes in Computer Science*, pages 101–115. Springer, November 2006.

Tim Berners-Lee. Semantic web road map, September 1998. Available at `http://www.w3.org/DesignIssues/Semantic.html`.

R. J. Brachman and J. G. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171–216, April 1985.

Leslie Carr, Sean Bechhofer, Carole Goble, and Wendy Hall. Conceptual Linking: Ontology-based Open Hypermedia. In *WWW10, Tenth World Wide Web Conference*, May 2001.

Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. Just the right amount: Extracting modules from ontologies. In *Proc. of the Sixteenth International World Wide Web Conference (WWW 2007)*, 2007a. URL `download/2007/CHKS07a.pdf`.

Bernardo Cuenca Grau, Yevgeny Kazakov, Ian Horrocks, and Ulrike Sattler. A logical framework for modular integration of ontologies. In *Proc. of the 20th Int. Joint Conf. on Artificial Intelligence (IJCAI 2007)*, 2007b. URL `download/2007/CKHS07a.pdf`.

S. Derriere, A. Richard, and A. Preite-Martinez. An ontology of astronomical object types for the virtual observatory. *Proc. of Special Session 3 of the 26th meeting of the IAU: Virtual Observatory in Action: New Science, New Technology, and Next Generation Facilities*, 2006.

Martin Dzbor, John Domingue, and Enrico Motta. Magpie – Towards a Semantic Web Browser. In Dieter Fensel, Katia Sycara, and John Mylopoulos, editors, *2nd*

International Semantic Web Conference, ISWC, volume 2870 of *Lecture Notes in Computer Science*. Springer, October 2003.

D. Fensel, F. van Harmelen, I. Horrocks, D. McGuinness, and P. F. Patel-Schneider. OIL: An ontology infrastructure for the semantic web. *IEEE Intelligent Systems*, 16(2):38–45, 2001. URL `download/2001/IEEE-IS01.pdf`.

Christine Golbreich, Songmao Zhang, and Olivier Bodenreider. The foundational model of anatomy in OWL: Experience and perspectives. *J. of Web Semantics*, 4(3), 2006.

John Goodwin. Experiences of using OWL at the ordnance survey. In *Proc. of the First OWL Experiences and Directions Workshop*, volume 188 of *CEUR Workshop Proceedings*. CEUR (`http://ceur-ws.org/`), 2005.

Volker Haarslev and Ralf Möller. RACER system description. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2001)*, volume 2083 of *Lecture Notes in Artificial Intelligence*, pages 701–705. Springer, 2001.

Siegfried Handschuh and Steffen Staab, editors. *Annotation for the Semantic Web*, volume 96 of *Frontiers in Artifical Intelligence and Applications*. IOS Press, 2003.

Simon Harper and Sean Bechhofer. Semantic Triage for Accessibility. *IBM Systems Journal*, 44(3):637–648, 2005.

Frank W. Hartel, Sherri de Coronado, Robert Dionne, Gilberto Fragoso, and Jennifer Golbeck. Modeling a description logic vocabulary for cancer research. *Journal of Biomedical Informatics*, 38(2):114–129, 2005.

Ian Horrocks and Ulrike Sattler. Decidability of $\mathcal{SHIQ}$ with complex role inclusion axioms. *Artificial Intelligence*, 160(1–2):79–104, December 2004.

Ian Horrocks and Ulrike Sattler. A tableaux decision procedure for $\mathcal{SHOIQ}$. In *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI 2005)*, pages 448–453, 2005. URL `download/2005/HoSa05a.pdf`.

Ian Horrocks, Peter F. Patel-Schneider, and Frank van Harmelen. Reviewing the design of DAML+OIL: An ontology language for the semantic web. In *Proc. of the 18th Nat. Conf. on Artificial Intelligence (AAAI 2002)*, pages 792–797. AAAI Press, 2002. ISBN 0-26251-129-0. URL `download/2002/AAAI02IHorrocks.pdf`.

Ian Horrocks, Peter F. Patel-Schneider, and Frank van Harmelen. From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language. *J. of Web Semantics*, 1 (1):7–26, 2003. ISSN 1570-8268. URL `download/2003/HoPH03a.pdf`.

Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The even more irresistible $\mathcal{SROIQ}$. In *Proc. of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 57–67. AAAI Press, 2006.

Ullrich Hustadt, Boris Motik, and Ulrike Sattler. Reducing SHIQ-description logic to disjunctive datalog programs. In *Proc. of the 9th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2004)*, pages 152–162, 2004.

A. Kalyanpur, Bijan Parsia, Evren Sirin, Bernardo Cuenca-Grau, and James Hendler. SWOOP: a web ontology editing browser. *J. of Web Semantics*, 4 (2), 2005a.

Aditya Kalyanpur, Bijan Parsia, Evren Sirin, and James Hendler. Debugging unsatisfiable classes in owl ontologies. *J. of Web Semantics*, 3(4):243–366, 2005b. URL `http://www.mindswap.org/papers/debugging-jws.pdf`.

Aaron Kershenbaum, Achille Fokoue, Chintan Patel, Christopher Welty, Edith Schonberg, James Cimino, Li Ma, Kavitha Srinivas, Robert Schloss, and J William Murdock. A view of OWL from the field: Use cases and experiences. In *Proc.*

of the Second OWL Experiences and Directions Workshop, volume 216 of *CEUR (http://ceur-ws.org/)*, 2006.

Holger Knublauch, Ray Fergerson, Natalya Noy, and Mark Musen. The Protégé OWL Plugin: An open development environment for semantic web applications. In Sheila A. McIlraith, Dimitris Plexousakis, and Frank van Harmelen, editors, *Proc. of the 2004 International Semantic Web Conference (ISWC 2004)*, number 3298 in Lecture Notes in Computer Science, pages 229–243. Springer, 2004. ISBN 3-540-23798-4.

Christos Kouroupetroglou, Michail Salampasis, and Athanasios Manitsaris. A semantic-web based framework for developing applications to improve accessibility in the www. In *W4A: Proceedings of the 2006 international cross-disciplinary workshop on Web accessibility (W4A)*, pages 98–108, New York, NY, USA, 2006. ACM Press. ISBN 1-59593-281-X. doi: http://doi.acm.org/10.1145/1133219.1133238.

Lee Lacy, Gabriel Aviles, Karen Fraser, William Gerber, Alice Mulvehill, and Robert Gaskill. Experiences using OWL in military applications. In *Proc. of the First OWL Experiences and Directions Workshop*, volume 188 of *CEUR Workshop Proceedings*. CEUR (`http://ceur-ws.org/`), 2005.

S. McIlraith, T. C. Son, and H. Zeng. Semantic web services. *IEEE Intelligent Systems*, 16:46–53, 2001.

Peter F. Patel-Schneider, Patrick Hayes, and Ian Horrocks. OWL Web Ontology Language semantics and abstract syntax. W3C Recommendation, 10 February 2004. Available at `http://www.w3.org/TR/owl-semantics/`.

Alan Rector and Jeremy Rogers. Ontological and practical issues in using a description logic to represent medical concept systems: Experience from GALEN. In *Reasoning Web, Second International Summer School, Tutorial Lectures*, volume 4126 of *LNCS*, pages 197–231. SV, 2006.

Alan Ruttenberg, Jonathan Rees, and Joanne Luciano. Experience using OWL DL for the exchange of biological pathway information. In *Proc. of the First OWL Experiences and Directions Workshop*, volume 188 of *CEUR Workshop Proceedings*. CEUR (`http://ceur-ws.org/`), 2005.

Lisa Seeman. The semantic web, web accessibility, and device independence. In *W4A '04: Proceedings of the 2004 international cross-disciplinary workshop on Web accessibility (W4A)*, pages 67–73, New York, NY, USA, 2004. ACM Press. ISBN 1-58113-903-9. doi: http://doi.acm.org/10.1145/990657.990669.

Amandeep Sidhu, Tharam Dillon, Elisabeth Chang, and Baldev Singh Sidhu. Protein ontology development using OWL. In *Proc. of the First OWL Experiences and Directions Workshop*, volume 188 of *CEUR Workshop Proceedings*. CEUR (`http://ceur-ws.org/`), 2005.

E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical owl-dl reasoner. URL `http://www.mindswap.org/papers/PelletJWS.pdf`. To appear, 2005.

Dagobert Soergel, Boris Lauser, Anita Liang, Frehiwot Fisseha, Johannes Keizer, and Stephen Katz. Reengineering thesauri for new applications: The AGROVOC example. *J. of Digital Information*, 4(4), 2004.

K. Spackman. Managing clinical terminology hierarchies using algorithmic calculation of subsumption: Experience with SNOMED-RT. *J. of the Amer. Med. Informatics Ass.*, 2000. Fall Symposium Special Issue.

SWEET. Semantic web for earth and environmental terminology (SWEET). Jet Propulsion Laboratory, California Institute of Technology, 2006. `http://sweet.jpl.nasa.gov/`.

Hironobu Takagi and Chieko Asakawa. Transcoding proxy for nonvisual web access. In *Proceedings of the Fourth International ACM Conference on Assistive Technologies*, pages 164–171. ACM Press, 2000.

Dmitry Tsarkov and Ian Horrocks. FaCT++ description logic reasoner: System description. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2006)*, volume 4130 of *Lecture Notes in Artificial Intelligence*, pages 292–297. Springer, 2006. URL `download/2006/TsHo06a.pdf`.

R. Volz, S. Handschuh, S. Staab, L. Stojanovic, and N. Stojanovic. Unveiling the hidden bride: Deep Annotation for Mapping and Migrating Legacy Data to the Semantic Web. *Journal of Web Semantics*, 2004.

W. A. Woods. What's in a link: Foundations for semantic networks. In Ronald J. Brachman and Hector J. Levesque, editors, *Readings in Knowledge Representation*, pages 217–241. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1985. ISBN 093461301X. Previously published in D. G. Bobrow and A. M. Collins, editors, *Representation and Understanding: Studies in Cognitive Science*, pages 35-82. New York Academic Press, 1975.

Yeliz Yesilada, Simon Harper, Carole Goble, and Robert Stevens. Dante annotation and transformation of web pages for visually impaired users. In *The Thirteenth International World Wide Web Conference*, 2004.

Yeliz Yesilada, Sean Bechhofer, and Bernard Horan. Personalised Dynamic Links on the Web. In *SMAP2006: 1st International Workshop on Semantic Media Adaptation and Personalization*, 2006.

Index