Gouveia, A. and Cardoso, J. OWL: Web Ontology Language. In Encyclopedia of Information Science and Technology, Information Science Reference, Pennsylvania, USA, 2007.

OWL – Web Ontology Language

Adélia Gouveia and Jorge Cardoso

Department of Mathematics and Engineering
University of Madeira
9050-390 Funchal, Portugal
Tef: +351 291 705 151
Fax: +351 291 705 199
jcardoso@uma.pt

INTRODUCTION

The World Wide Web (WWW) emerged in 1989, developed by Tim Berners-Lee who proposed to build a system for sharing information among physicists of the CERN (*Conseil Européen pour la Recherche Nucléaire*), the world's largest particle physics laboratory.

Currently, the WWW is primarily composed of documents written in HTML (Hyper Text Markup Language), a language that is useful for visual presentation (Cardoso and Sheth 2005). HTML is a set of "markup" symbols contained in a Web page intended for display on a Web browser. Most of the information on the Web is designed only for human consumption. Humans can read Web pages and understand them, but their inherent meaning is not shown in a way that allows their interpretation by computers (Cardoso and Sheth 2006).

Since the visual Web does not allow computers to understand the meaning of Web pages (Cardoso 2007), the W3C (World Wide Web Consortium) started to work on a concept of the Semantic Web with the objective of developing approaches and solutions for data integration and interoperability purpose. The goal was to develop ways to allow computers to understand Web information.

The aim of this chapter is to present the Web Ontology Language (OWL) which can be used to develop Semantic Web applications that understand information and data on the Web. This language was proposed by the W3C and was designed for publishing, sharing data and automating data understood by computers using ontologies. To fully comprehend OWL we need first to study its origin and the basic blocks of the language. Therefore, we will start by briefly introducing XML (eXtensible Markup Language), RDF (Resource Description Framework), and RDF Schema (RDFS). These concepts are important since OWL is written in XML and is an extension of RDF and RDFS.

BACKGROUND

Everyday, the Web becomes more attractive as an information sharing infrastructure. However, the vast quantity of data made available (for example, Google indexes more than 13 billion pages) makes it difficult to find and access the information required by the wide diversity of users. This limitation arises because most documents on the Web are written in HTML (HTML 2007), a language that is useful for visual presentation but which is semantically limited. As a result, humans can read and understand HTML Web

pages, but the contents of Web pages are not defined in a way that computers can understand them. If computers are not able to understand the content of Web pages it becomes impossible to develop sophisticated solutions to enable the interoperability and integration between systems and applications.

The aim of the Semantic Web is to make the information on the Web understandable and useful to computer applications and in addition to humans. *"The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation"* (Berners-Lee, Hendler et al. 2001). The Semantic Web is a vision for the future of the Web, in which information is given explicit meaning, making it easier for machines to automatically process and integrate the information available on the Web.

One of the corner stones of the Semantic Web is the OWL. OWL provides a language that can be used by/on applications that need to understand the meaning of information instead of just parsing data for display purposes. Nowadays, several projects already rely on semantics to implement their applications. Example include semantic wikis (Campanini, Castagna et al. 2004), social networks (Ding, Finin et al. 2005), semantic Blogs (Cayzer and Shabajee 2003), and semantic web services (McIlraith, Son et al. 2001),

THE SEMANTIC WEB STACK

The Semantic Web identifies a set of technologies and standards which form the basic building blocks of an infrastructure that supports the vision of the Web associated with meaning. Figure 1 illustrates the different parts of the Semantic Web architecture. It starts with the foundation of URI (Universal Resource Identifier) and Unicode. URI is a formatted string that serves as a means of identifying abstract or physical resources. For example, http://dme.uma.pt/jcardoso/index.htm identifies the location from where a Web page can be retrieved and urn:isbn:3-540-24328-3 identifies a book using its ISBN. Unicode provides a unique number for every character, independent of the underlying platform, program, or language.

Directly above URI and Unicode we find the syntactic interoperability layer in the form of XML, which in turn underlies RDF and RDFS. Web ontology languages are built on top of RDF and RDFS. The last three layers are logic, proof, and trust, which have not been significantly explored. Some of the layers rely on the digital signature component to ensure security.

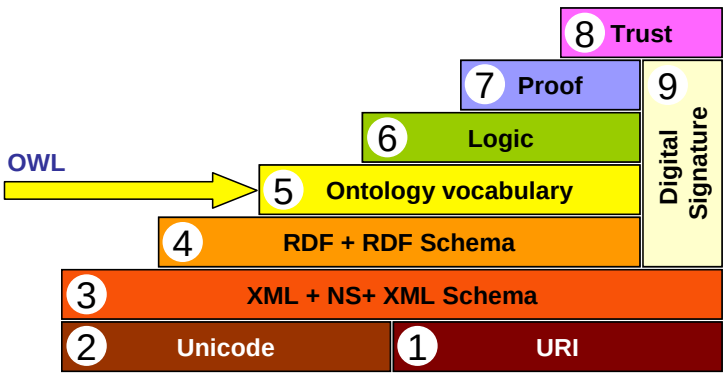

FIGURE 1. Semantic Web layered architecture (Berners-Lee, Hendler et al. 2001)

In the following sections we briefly describe the most relevant layers (XML, RDF, and RDFS). While the notions presented have been simplified, they give a reasonable conceptualization of the various components of the Semantic Web.

XML

The eXtensible Markup Language (XML) (Decker, Melnik et al. 2000; XML 2007) was originally pictured as a language for defining new document formats for the WWW. An important feature of this language is the separation of content from presentation, which makes it easier to select and/or reformat the data. SGML (Standard Generalized Markup Language) and XML are text-based formats that provide mechanisms for describing document structures using markup tags (words surrounded by '<' and '>'). Both HTML and XML representations use tags such as <h1> or <name>, and information between those tags, referred to as the content of the tag. However, there are significant differences between HTML and XML. XML is case sensitive while HTML is not. This means that in XML the start tags <Table> and <table> are different, while in HTML they are the same. Another difference is that HTML has predefined elements and attributes whose behavior is well specified, while XML does not. Instead, users can create their own XML vocabularies that are specific to their application or business' needs.

The following structure shows an example of an XML document identifying a 'Contact' resource. The document includes various metadata markup tags, such as <first_name>, <last_name>, and <email>, which provides various details about a contact.

```
<Contact contact_id="1234">
    <first_name> Jorge </first_name>
    <last_name> Cardoso </last_name>
    <organization> University of Madeira </organization>
    <email> CARDOSO@UMA.PT </email>
    <phone> +51 291 705 156 </phone>
</Contact>
```

While XML has gained much of the world's awareness, it is significant to identify that XML is simply a way of standardizing data formats. But from the point of view of semantic interoperability, XML has restrictions. One important characteristic is that there is no way to recognize the semantics of a particular domain because XML aims at a document structure and enforces no common interpretation of the data. Although XML is simply a data-format standard, it is part of a set of technologies that constitute the foundations of the Semantic Web.

RDF

Resource Description Framework (RDF) (RDF 2002), was developed by the W3C to provide a common way to describe information so it could be read and understood by computer applications. RDF was designed using XML as the underlying syntax language. RDF provides a model for describing resources on the Web. A resource is an element (document, Web page, printer, user, etc) on the Web that is uniquely identifiable by a URI. The RDF model is based upon the idea of making statements

about resources in the form of a subject-predicate-object expression, a ‘triple’ in RDF terminology.

- Subject is the resource, i.e. the thing that is being described;
- Predicates are aspects about a resource, and expresses the relationship between the subject and the object;
- Object is the value that is assigned to the predicate.

RDF has a very limited set of syntactic constructs, no other constructs except for triples is allowed. Every RDF document is equivalent to an unordered set of triples. Let us write a RDF triple that describes the following statement:

“The creator of the page named ComputersToday is John.”

In this example, ‘http://www.semanticweb.pt/ComputersToday’ is a resource, and it has a property, ‘Creator’, with the value ‘John’. The resulting RDF statement is:

st = (http://www.semanticweb.pt/ComputersToday, Creator, John)

The statement can also be graphically represented as illustrated in Figure 2.

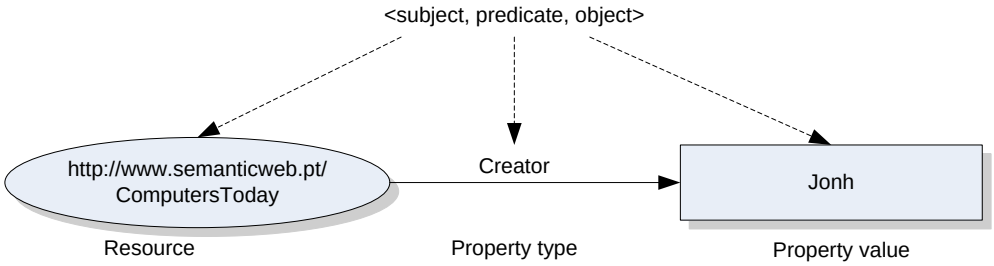

FIGURE 2 - RDF GRAPH

One way to represent the statement in Figure 2 using RDF language is the following:

```
<? xml version="1.0" ?>
<RDF xmlns = "http://w3.org/TR/1999/PR-rdf-syntax-19990105#"
     xmlns:DC = "http://dublincore.org/2003/03/24/dces#">

    <Description about =
              "http://www.semanticweb.pt/ComputersToday">
    <DC:Creator>John</DC:Creator>
  </Description>
</RDF>
```

The first lines of this example use namespaces to explicitly define the meaning of the notions that are used. The first namespace xmlns:rdf=”http://w3.org/TR/1999/PR-rdf-syntax-19990105#” refers to the document describing the syntax of RDF. The second namespace http://dublincore.org/2003/03/24/dces# refers to the description of the Dublin Core (DC) (DC 2005), a basic ontology about authors and publications.

RDF Schema

RDF Schema (RDFS) (XMLSchema 2005) is technologically more advanced when compared to RDF. RDFS describes the resources with classes, properties, and values. RDFS associates the resources in classes and states the relations between these classes, or declares properties and specifies the domain and range of these properties. RDFS' specification consists of some basic classes and properties that can be extended to any given domain.

Classes in RDFS are much like classes in object oriented programming languages. These allows resources to be defined as instances of classes, and subclasses of classes. Properties can be seen as attributes that are used to describe the resources by assigning values to them. RDF is used to declare a property and RDFS can extend this capability by defining the domain and the range of that property. (However, RDFS has some limitations but these have been resolved with the introduction of OWL.)

THE WEB ONTOLOGY LANGUAGE

The Web Ontology Language (OWL) (OWL 2004) is one of the most important ontology languages. It enables the interoperability of applications and allows computers to understand the Web's content. In this respect it is more expressive than XML, RDF or RDF Schema due to providing additional vocabulary along with formal semantics.

OWL Flavors

There are three OWL sublanguages: OWL Lite, OWL DL, and OWL Full. An important feature of each sublanguage is its expressiveness. OWL Lite is the least expressive and the OWL Full is the most expressive sublanguage. OWL DL is more expressive that OWL Lite but less expressive than OWL Full. In other words, this entails that every legal OWL Lite ontology is a legal OWL DL ontology; every legal OWL DL ontology is a legal OWL Full ontology; every valid OWL Lite conclusion is a valid OWL DL conclusion; and every valid OWL DL conclusion is a valid OWL Full conclusion.

OWL Full is the most expressive of the OWL sublanguages and it uses the entire OWL language primitives. It is intended to be used in situations where very high expressiveness is more important than being able to guarantee the decidability or computational completeness of the language. This sublanguage is meant for users who want maximum expressiveness and the syntactic freedom of RDF, but with no computational guarantees.

OWL DL is a sublanguage of OWL Full that restricts the application of OWL and RDF constructors. OWL DL (DL stands for Description Logics) is not compatible with RDF, in the same way that not every RDF document is a legal OWL DL document, although every legal OWL DL document is a legal RDF document. This sublanguage supports those users who want the maximum expressiveness without losing computational completeness and decidability.

OWL Lite is syntactically the simplest sublanguage. It is intended to be used in situations where only a simple class hierarchy and constraints are needed. This sublanguage supports those users primarily needing a simple classification hierarchy and constraint features.

Note that every OWL Lite ontology or conclusion is a legal OWL DL ontology or conclusion, but not the inverse, and so on for OWL DL and OWL Full, as showed in the following figure:

FIGURE 3- OWL SUBLANGUAGES

The choice between OWL Lite and OWL DL may be based upon whether the simple constructs of OWL Lite are sufficient or not. The choice between OWL DL and OWL Full may be based upon whether it is important to be able to carry out automated reasoning on the ontology or whether it is important to be able to use highly expressive and powerful modeling facilities.

OWL syntax

In this section we describe the syntax of OWL. We illustrate step-by-step how to build an ontology using OWL. We also explain how to define the heather of ontology, its classes, properties and relationships. After reading this section the reader should be able to recognize an ontology written in OWL and identify some of its components.

Header

The first element in an OWL document is an rtf:RDF element which specifies a set of XML namespace's declarations that provide a means to unambiguously interpret identifiers and make the rest of the ontology presentation much more readable. For example,

```
<rdf:RDF
  xmlns="http://apus.uma.pt/~adelia/RUD.owl#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xml:base="http://apus.uma.pt/~adelia/RUD.owl">
```

A namespace is composed by: reserved XML attribute xmlns, a prefix that identify the namespace and the value.

Information version

After the namespace declaration, an OWL document specifies a collection of assertions that are grouped under an owl:ontology element and offers details about the ontology:

- owl:versionInfo – provides information about the current ontology.
- owl:priorVersion – indicates an earlier version of the current one.
- owl:backwardCompatibleWith – contains a reference to an ontology that is a prior version of the containing ontology that is backward compatible with it.
- owl:incompatibleWith – indicates that the containing ontology is not backward compatible with the referenced ontology.
- owl:imports – only this assertion has a formal meaning to the ontology and represents a set of other ontologies that are considered to be part of the current ontology. Note that owl:imports is a transitive property because if ontology A imports ontology B, and ontology B imports ontology C, then ontology A also imports ontology C.

The following is a simple example:

```
<rdf:RDF>
...
<owl:Ontology rdf:about="">
    <rdfs:comment> University Ontology </rdfs:comment>
    <owl:versionInfo> v.1 2006-9-05 </owl:versionInfo>
    <owl:priorVersion>
      <owl:Ontology rdf:about = "http://apus.uma.pt/~adelia/RUD.owl"/>
    </owl:priorVersion>
    <rdfs:label> University Ontology </rdfs:label>
</owl:Ontology>
...
</rdf:RDF>
```

Class element

Classes are a collection of individuals, a way of describing part of the world. They are defined in an OWL document with the owl:Class element. For example, the class “Teacher” can be define as follows,

```
<owl:Class rdf:ID="Teacher">
    <rdfs:subClassOf rdf:resource="#Person"/>
</owl:Class>
```

Note that the rdf:ID element defines the name of the class. If we want to make reference to a class we use the rdf:resource element. An OWL ontology can represent the hierarchy between classes using the element owl:subClassOf. For example, the class “Teacher” is a subclass of “Person”.

Between the two classes it is possible to establish relations using owl:equivalentClass and owl:disjoinWith elements. The assertion owl:equivalentClass when applied to two classes A and B, represents that class A has the same individuals as

class B. For example, the class “faculty” is equivalent to the “academicStaffMember” class:

```
<owl:Class rdf:ID=”faculty”>
    <owl:equivalentClass rdf:resource="#academicStaffMember"/>
</owl:Class>
```

The owl:disjoinWith element applied on two classes A and B suggest that class A and B disjoin, i.e., if an instance is member of class A it cannot be an instance of class B. For example, a “Full Professor” cannot be an “Associate Professor” at the same time.

```
<owl:Class rdf:about=”#AssociateProfessor”>
    <owl:disjoinWith rdf:resource="#FullProfessor"/>
</owl:Class>
```

Complex class

Another way to create classes in OWL is to combine simple classes using Boolean operators (union, intersection, and complement) and create complex classes. The members of the class are completely specified by the Boolean operators. The owl:unionOf element applied on classes A and B creates a new class that contains all members from class A and B. For example, the combination of the class “staff members” and the class “student” create the new class “peopleAtUni”, as shown below,

```
<owl:Class rdf:ID="peopleAtUni">
   <owl:unionOf rdf:parseType="Collection">
      <owl:Class rdf:about="#staffMember"/>
      <owl:Class rdf:about="#student"/>
   </owl:unionOf>
</owl:Class>
```

The owl:intersectionOf element creates a new class from the two classes A and B which has elements that were both in class A and class B, which follows as,

```
<owl:Class rdf:ID="facultyInDME">
   <owl:intersectionOf rdf:parseType="Collection">
     <owl:Class rdf:about="#faculty"/>
     <owl:Restriction>
        <owl:onProperty rdf:resource="#belongsTo"/>
        <owl:hasValue rdf:resource="#DMEDepartment"/>
     </owl:Restriction>
   </owl:intersectionOf>
</owl:Class>
```

The individuals of the new class created in this example are those individuals that are members of both the classes “faculty” and the anonymous class created by the restriction on the property “belongTo”.

The owl:complement element selects all individuals from the domain that do not belong to a certain class,

```
<owl:Class rdf:about="#course">
   <rdfs:subClassOf>
      <owl:Class>
          <owl:complementOf rdf:resource="#staffMember"/>
      </owl:Class>
   </rdfs:subClassOf>
</owl:Class>
```

In this example, the class “course” has as its members all individuals that do not belong to the” staffMember” class.

Property

Properties let us describe a kind of relationship between members of classes. In an OWL document two types of properties are distinguished:

- Object properties which relate objects to other objects, i.e. instances of a class with instances of another class. In the next example the object property “isTaughtBy” relates the class “course” with the class “academicsStaffMember”. This means that a “course” “isTaughtBy” an instance of the “academicStaffMember” class.

```
<owl:ObjectProperty rdf:ID="isTaughtBy">
   <owl:domain rdf:resource="#course"/>
   <owl:range rdf:resource="#academicStaffMember"/>
</owl:ObjectProperty>
```

- Datatype property which relates objects to data type values. OWL does not have predefined data types, but it allows one to use the XML Schema data types. In following example, the year in which a tourist was born is specified using the “http://www.w3.org/2001/XLMSchema#nonNegativeInteger” data type from the XML Shema.

```
<owl:DatatypeProperty rdf:ID="ageYear">
   <rdfs:domain rdf:resource="#Person"/>
   <rdfs:range rdf:resource=
http://www.w3.org/2001/XLMSchema#nonNegativeInteger"/>
</owl:DatatypeProperty>
```

Note that both kinds of properties can use the rdfs:domain and rdfs:range element to restrict the relation.

Property restrictions

More elaborate boundaries can be made by applying restrictions to a property, this results in the subclasses of individuals that satisfy that condition. There are two kinds of restrictions: values constraints and cardinality constraints. Examples of values constraints include owl:allValuesFrom, owl:someValuesFrom, and owl:hasValues.

owl:allValuesFrom: Defines the set of individuals, for which all the values of the restricted property are instance of a certain class:

```
<owl:Class rdf:about="#firstYearCourse">
   <rdfs:subClassOf>
     <owl:Restriction>
       <owl:onProperty rdf:resource="#isTaughtBy"/>
       <owl:allValuesFrom rdf:resource="#professor"/>
     </owl:Restriction>
   </rdfs:subClassOf>
</owl:Class>
```

In this example, the individuals that are members of the class “firstYearCourse” are all the courses that have the property “isTaughtBy” assigned to a “professor”

owl:someValuesFrom: Defines the set of individuals that have at least one relation with an instance of a certain class, for example:

```
<owl:Class rdf:about="#academicStaffMember">
   <rdfs:subClassOf>
     <owl:Restriction>
       <owl:onProperty rdf:resource="#teaches"/>
       <owl:someValuesFrom rdf:resource="#undergraduateCourse"/>
      </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

owl:hasValues: Defines a set of individuals for which the value of the restricted property is equal to a certain instance. For example the individuals of the class “mathCourse” can be characterized as those that are taught by the professor “949352”:

```
<owl:Class rdf:about="#mathCourse">
   <rdfs:subClassOf>
     <owl:Restriction>
       <owl:onProperty rdf:resource="#isTaughtBy"/>
       <owl:hasValues rdf:resource="#949352"/>
   </owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
```

Cardinality constraints point out how many times the property can be used on an instance. Examples include owl:maxCardinality, owl:minCardinality, and owl:cardinality.

owl:maxCardinality: Defines the set of individuals that have at the most N distinct values of the property concerned. For example, we can specify that the class “department” has at the most 30 members:

```
<owl:Class rdf:about="#department">
   <rdfs:subClassOf>
```

```
      <owl:Restriction>
        <owl:onProperty rdf:resource="#hasMember"/>
        <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger">
          30
         </owl:maxCardinality>
      </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

owl:minCardinality: Defines the set of individuals that have at least N distinct values of the property concerned. For example, a course must be taught at least by one teacher. In OWL it is defined as follows:

```
<owl:Class rdf:about="#course">
   <rdfs:subClassOf>
     <owl:Restriction>
       <owl:onProperty rdf:resource="#isTaughtBy"/>
       <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">
         1
        </owl:minCardinality>
     </owl:Restriction>
   </rdfs:subClassOf>
</owl:Class>
```

owl:cardinality: Defines the set of individuals that have an exact number of distinct values of the property concerned. This element is used to specify a precise number, i.e. to express that a property has a minimum cardinality which is equal to the maximum cardinality.

Properties' Characteristics

Properties' characteristics add more expressivity to the OWL language. owl:equivalentProperty and owl:inverseOf elements are examples of those characteristics. The owl:equivalentProperty element associate properties that have the same range and the same domain. For example, the property "leacturesIn" is equivalent to "teaches" and in OWL this can be represented as:

```
<owl:ObjectProperty rdf:ID="leacturesIn">
    <owl:equivalentProperty rdf:resource="#teaches"/>
</owl:ObjectProperty>
```

The owl:inverseOf element can be used to define inverse relation between properties. If property P' is stated to be the inverse of property P'', then if X'' is related to Y'' by the P'' property, then Y'' is related to X'' by the P' property. For example, "teacher teaches a course" is the inverse of "a course is taught by a teacher". This can be expressed in OWL as:

```
<owl:ObjectProperty rdf:resource="#teaches">
    <owl:inverseOf rdf:resource="#isTaughtBy"/>
</owl:ObjectProperty>
```

The property element has some properties that can be defined directly:

- Function property – The function property (owl:FunctionProperty) defines a property that has at the most one value for each instance.

- InverseFuncionalProperty – In OWL, by using the InverseFuncionalProperty (owl:InverseFuncionalProperty) it is possible to define properties that have different values to different instances, i.e. two different instances can not have the same values.
- Transitive property – The transitive property is understood as: if the pair (x, y) is an instance of the transitive property P, and the pair (y, z) is an instance of P, we can infer the pair (x, z) is also an instance of P.
- Symmetric property – The symmetric property (owl:SymmetricProperty) is interpreted as follows: if the pair (x, y) is an instance of A, then the pair (y, x) is also an instance of A.

The following example illustrate the application of the owl:SymmetricProperty and owl:TransitiveProperty elements.

```
<owl:ObjectProperty rdf:ID="hasSameGradeAs">
   <rdf:type rdf:resource="&owl;TransitiveProperty"/>
   <rdf:type rdf:resource="&owl;SymmetricProperty"/>
   <rdfs:domain rdf:resource="#student"/>
   <rdfs:range rdf:resource="#student"/>
</owl:ObjectProperty>
```

FUTURE TRENDS

Many researchers worldwide have recognized that the Semantic Web (or Web3.0) is the key to develop the new generation of information systems. The number of international conferences organized every year on this topic clearly shows the interest and importance of this new technology. In this context, OWL is the most widespread language to develop a new breed of the Semantic Web-based applications. OWL has been used in many areas; some applications and tools use this conceptual approach to build Semantic Web based systems. According to TopQuadrant (TopQuadrant 2005), a consulting firm that specializes in Semantic Web technologies, the market for semantic technologies will grow at an annual growth rate of between 60% and 70% until 2010

In the near future, we will see the use of OWL to implement applications raging from semantic social networking, semantic RSS, semantic podcasts, semantic wikis, semantic blogs to semantic mashups. As you can see, we will be devising a solution that matches most of executives' future product acquisitions strategies. Adding semantics to these types of applications is important since in a survey of 8,300 executives from McKinsey it was found that when asked about their plans to invest in tools in the future, the answers given included those applications.

Enterprise Information Integration (EII) is another area that will benefit from the Semantic Web and OWL. Today, integration is a top priority for many European and worldwide enterprises. Most organizations have already realized that the use of Semantic Web technologies (Berners-Lee, Hendler et al. 2001) is the best solution to support cross-organizational cooperation for SMEs that operate in dynamically changing work environments. Semantic Web technologies are already viewed as a key technology to resolve the problems of interoperability and integration within the heterogeneous world of ubiquitously interconnected systems with respect to the nature of components, standards, data formats, protocols, etc. Moreover, we also believe that Semantic Web technologies can facilitate not only the discovery of heterogeneous components and data integration, but also the communication, coordination and

collaboration behavioral of employees and individuals. Semantics can help not only the system, but also human integration and interoperability.

Managing information in enterprises faces three barriers that have to be overcome: the diverse data formats, the disparate nature of content and the need to drive "intelligence" from this content. The Semantic Web helps to surpass these limitations by providing a way to add semantic metadata to documents. Metadata allows software programs to automatically understand the full context and meaning of each document. So it is accurate to say that semantics will enable information integration and analyses in the following tasks:

- Extracting, organizing and standardizing information from many disparate and heterogeneous content sources and formats.
- Identifying interesting and relevant knowledge from heterogeneous sources and formats.
- Making efficient use of the extracted knowledge and content by providing tools that enables fast and high-quality querying, browsing and analysis of relevant and actionable information.

Finally, programming the Semantic Web with OWL can reduce and eliminate terminological and conceptual confusion by defining a shared understanding, that is, a unifying framework enabling communication and cooperation amongst people in reaching a better inter-enterprise organization. Presently, one of the most important roles ontology plays in communication is that it provides unambiguous definitions for terms used in a software system, but semantics needs to be applied rapidly to human integration to enable communication, coordination, and cooperation. The use of ontologies for improving communication has already been shown to work in practice.

CONCLUSIONS

The Semantic Web is the future vision of the current Web, where information will have a precise meaning. Currently, the WWW is primarily composed of documents written in a language (HTML) that is useful for visual presentation, but not for computerized understanding. The Semantic Web is not a separate Web but an extension of the current one, in which information is a given well-defined meaning, enabling computers and people to work better in cooperation. To make possible the creation of the Semantic Web it is important to have a language that: (1) describes the concepts of a given domain and (2) creates ontologies. One of the most prominent ontology languages to achieve those two tasks is OWL (Ontology Web Language) which can be used to develop Semantic Web applications. These applications will constitute a new wave of enhanced systems that will understand better the domain in which they are working and with which they interact. OWL defines a common set of terms that are used to describe and represent a specific domain. Thus, standard OWL enables the Web to be a global infrastructure for sharing both documents and data, which makes searching and reusing information easier and more reliable as well. OWL can be used by applications to improve search engines on the Web and tools to manage knowledge. In this chapter we have laid out the foundations of the Semantic Web, its associated languages and standards. These elements are the basic building blocks of any Semantic Web application.

REFERENCES

Berners-Lee, T., J. Hendler, et al. (2001). The Semantic Web. Scientific American. May 2001.

Campanini, S. E., P. Castagna, et al. (2004). Platypus wiki: a semantic wiki wiki web. In Semantic Web Applications and Perspectives. Proceedings of 1st Italian Semantic Web Workshop.

Cardoso, J. (2007). Semantic Web Services: Theory, Tools and Applications. New York, NY, USA, IGI Global, ISBN:978-1-59904-045-5.

Cardoso, J. and A. Sheth (2005). Semantic Web Process: powering next generation of processes with Semantics and Web services. Heidelberg, Lecture Notes in Computer Science, Springer-Verlag, Vol. 3387, ISBN:3-540-24328-3.

Cardoso, J. and A. Sheth (2006). Semantic Web Services, Processes and Applications, Springer, ISBN:0-38730239-5.

Cayzer, S. and P. Shabajee (2003). Semantic Blogging and Bibliography Management Blogtalk the First European Conference on Weblogs (Blogtalk 2003) Vienna, Austria.

DC. (2005). "The Dublin Core Metadata Initiative." Retrieved 9 May 2007, from http://dublincore.org/.

Decker, S., S. Melnik, et al. (2000). "The Semantic Web: The Roles of XML and RDF." Internet Computing 4(5): 63-74.

Ding, L., T. Finin, et al. (2005). "Analyzing Social Networks on the Semantic Web." IEEE Intelligent Systems 1(9).

HTML. (2007). "Hyper Text Markup Language." Retrieved 9 May 2007, from http://www.w3.org/html/.

McIlraith, S., T. C. Son, et al. (2001). "Semantic Web Services." IEEE Intelligent Systems 16(2): 46-53.

OWL. (2004). "Web Ontology Language (OWL)." Retrieved 9 May 2007, from http://www.w3.org/TR/owl-features/.

RDF. (2002). "Resource Description Framework (RDF)." Retrieved 9 May 2007, from http://www.w3.org/RDF/.

TopQuadrant. (2005). "TopQuadrant." 2005.

XML. (2007). "Extensible Markup Language (XML)." Retrieved 9 May 2007, from http://www.w3.org/XML/.

XMLSchema. (2005). "XML Schema." Retrieved 9 May 2007, from http://www.w3.org/XML/Schema.

TERMS AND DEFINITIONS

Metadata – Metadata is data that describe other data. Generally, a set of metadata describes a single set of data, called a resource.

Ontology – Is a description of concepts and relationships that can be used by people or software agents that want to share information within a domain. An ontology document defines the terms used to describe and represent a domain.

OWL – Is a markup language for publishing and sharing data using ontologies on the Internet. OWL is a vocabulary extension of the RDF and is derived from the DAML+OIL Web Ontology Language.

RDF – Resource Description Framework is a family of World Wide Web Consortium (W3C) specifications originally designed as a metadata model using XML but which has come to be used as a general method of modeling knowledge, through a variety of syntax formats.

RDFS – RDF Schema is an extensible knowledge representation language, providing basic elements for the definition of ontologies, otherwise called RDF vocabularies, intended to structure RDF resources.

Semantic Web – The Semantic Web provides a common framework that allows data to be shared and reused across applications, enterprises, and community boundaries. It is a collaborative effort led by W3C with the participation of a large number of researchers and industrial partners.

XML – The eXtensible Markup Language (XML) is a simple, very flexible text format derived from SGML (ISO 8879). XML is accepted as a standard for data interchanged on the Web, allowing for the structuring of data but without meaning.