Chapter 6
Tools for Ontology Engineering and Management

Lambrini Seremeti and Achilles Kameas

6.1 Introduction

Nowadays, ontologies are developed for and used in such diverse fields as qualitative modeling, language engineering, database design, information retrieval and extraction, knowledge management and organization, search and retrieval of information, e-commerce and configuration. The engineering part of developing ontologies comprises a complex set of activities that are conducted during conceptualization, design, implementation and deployment of ontologies. Ontology engineering covers a whole range of topics and issues, such as philosophical and metaphysical issues and knowledge representation formalisms, methodology of ontology development, recent Web technologies such as XML, RDF, OWL and their derivatives, business process modeling, common sense knowledge, systematization of domain knowledge, Internet information retrieval, standardization, evaluation, ontology integration with agents and applications, etc. It also helps in defining the essential concepts of the world of interest, provides design rationale for a knowledge base, supports a more disciplined design of a knowledge base and enables the accumulation of knowledge about it. As a consequence, the use of specific software tools that enable ontology conceptualization, representation, construction and use becomes an important aspect of building ontologies.

In this chapter, we first discuss a classification of ontology development and management tools (Section 6.2) according to the tasks that involve an ontology, for which we also provide some representative tools as examples. In Section 6.3, we discuss issues to consider when selecting an appropriate tool. Finally, Section 6.4 concludes this chapter with future, trends on tool, for ontology construction and evolution.

A. Kameas (✉)
Distributed Ambient Information Systems Group, Hellenic Open University & Computer Technology Institute, Patras, Hellas Greece
e-mail: kameas@eap.gr

R. Poli et al. (eds.), *Theory and Applications of Ontology: Computer Applications*, DOI 10.1007/978-90-481-8847-5_6, © Springer Science+Business Media B.V. 2010

6.2 Classification of Ontology Tools

The rapid growth of documents, web pages and other types of textual content pose a great challenge to modern content management systems. Ontologies offer an efficient way to reduce information overload by encoding the structure of a specific domain thus offering easier access to the information. There are numerous ontology engineering and management tools in use today. Most of them have resulted from efforts of research groups and university labs, so they are currently free. So far, there were several efforts to develop a comprehensive classification of ontology tools. One widely adopted taxonomy has been proposed by the OntoWeb Consortium (OntoWeb Consortium, 2002) and includes the following large categories: Ontology development tools, Ontology merge and integration tools, Ontology evaluation tools, Ontology-based annotation tools, Ontology storage and querying tools, and Ontology learning tools. In this chapter, we shall adopt a shallow classification of ontology tools in two large categories: specialized ontology engineering tools and integrated ontology engineering environments (see Fig. 6.1).

6.2.1 Specialized Ontology Engineering Tools

These are the tools that support a restricted set of activities of the entire ontology life-cycle (design, deployment, maintenance, evolution). They can be divided in ontology engineering tools, ontologies combination tools, and ontology management tools.

6.2.1.1 Ontology Engineering Tools

Ontology engineering is a set of tasks related to the development of ontologies for a particular domain. It aims at making explicit the knowledge contained within software applications, and within enterprises and business procedures for this domain. An ontology engineering tool is any tool used for creating ontologies or similar semantic documents. From this perspective, these tools can be classified in ontology building tools and ontology learning tools. The former help engineers construct an ontology from scratch, whereas the latter can be used for the (semi)automatic construction of an ontology.

Ontology Building Tools

An ontology building process includes problem specification, domain knowledge acquisition and analysis, conceptual design and commitment to community ontologies, iterative construction and testing, publishing the ontology as terminology, and possibly populating a conforming knowledge base with ontology individuals. While the process may be strictly a manual exercise, there are tools available that can automate portions of it. Some examples of the so-called ontology editors are CODE (http://www.cs.indiana.edu/~treicche/code.pdf), DOE

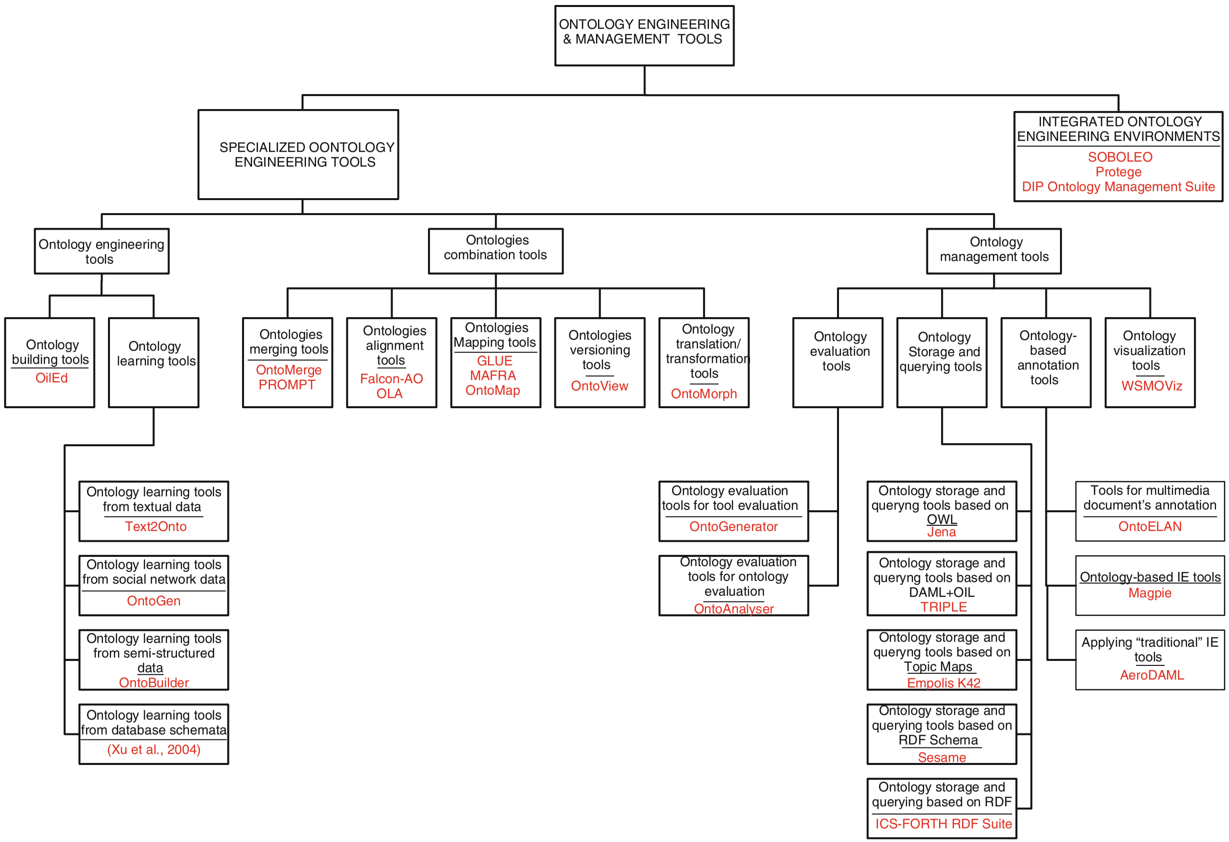

Fig 1. Taxonomy of ontology engineering and management tools and environments

(http://homepages.cwi.nl/~troncy/DOE), DUET (http://codip.grci.com/Tools/Tools.html), JOE (http://www.engr.sc.edu/research/CIT/demos/java/joe), ODE (http://www.swi.psy.uva.nl/wondertools/html/ODE.html), OilEd (http://oiled.man.ac.uk) and OntoSaurus (http://www.isi.edu/isd/ontosaurus.html).

OilEd (http://www.redland.opensource.ac.uk/demo/) is a graphical ontology editor developed by the University of Manchester that allows the user to build ontologies using DAML+OIL. The initial intention behind OilEd was to provide a simple editor that demonstrates the use of OIL language. Its current versions do not provide a full ontology development environment – it does not actively support the development of large–scale ontologies, the migration and integration of ontologies, versioning, argumentation and many other activities that are involved in ontology construction. Rather, it is the "NotePad" of ontology editors, offering enough functionality to allow users to build ontologies and to demonstrate how one can use the FaCT reasoner to check those ontologies for consistency.

Ontology Learning Tools

Ontology learning is a wide domain of research that consists of ontology enrichment, inconsistency resolution and ontology population. Ontology enrichment is the task of extending an existing ontology with additional concepts and relations and placing them in the correct position in the ontology. Inconsistency resolution is the task of resolving inconsistencies that appear in an ontology with the view to acquire a consistent (sub)ontology. Ontology population is the task of adding new instances of concepts into the ontology.

Acquiring domain knowledge for building ontologies requires much time and many resources. In this sense, one can define ontology learning as the set of methods and techniques used for building an ontology from scratch, or enriching, or adapting an existing ontology in a semi-automatic fashion using several sources. Several approaches exist for the partial automatization of the knowledge acquisition process. To carry out this automatization, natural language analysis and machine learning techniques can be used. Alexander Maedche and Steffen Staab (2001) distinguish different ontology learning approaches focused on the type of input used for learning. In this sense, they propose the following classification: ontology learning from text, from dictionary, from knowledge base, from semi-srtuctured schemata and from relational schemata.

Depending on the different assumptions regarding the provided input data, ontology learning can be addressed via different tasks: learning just the ontology concepts, learning just the ontology relationships between the existing concepts, learning both the concepts and relations at the same time, populating an existing ontology/structure, dealing with dynamic data streams, simultaneous construction of ontologies giving different views on the same data, etc. More formally, the ontology learning tasks are defined in terms of mappings between ontology components, where some of the components are given and some are missing and one wants to induce the missing ones. We shall use a different classification of ontology learning tools, that is based on the input data provided.

Ontology Learning Tools from Textual Data

As human language is a primary means of knowledge transfer, ontology learning from relevant text collections has been among the most successful strategies towards developing and maintaining ontologies dynamically. More information on ontology learning from text can be found in Buitelaar et al. (2005). Some examples of systems for ontology learning from text are KEA (Jones and Paynter, 2002), OntoLearn (Velardi et al., 2005), Welkin (Alfonseca and Rodriguez, 2002), and Text2Onto (Ciniamo and Volker, 2005).

Text2Onto (Ciniamo and Volker, 2005) is a framework for ontology learning from textual resources. Three main features distinguish Text2Onto from its earlier framework TextToOnto, as well as other state-of-the-art ontology learning frameworks. First, by representing the learned knowledge at a meta-level in the form of instantiated modeling primitives within a so called Probabilistic Ontology Model (POM), it remains independent of a concrete target language, while being able to translate the instantiated primitives into any (reasonably expressive) knowledge representation formalism. Second, user interaction is a core aspect of Text2Onto and the fact that the system calculates a confidence for each learned object, allows designing sophisticated visualizations of the POM. Third, by incorporating strategies for data-driven change discovery, it avoids processing the whole corpus from scratch each time it changes, only selectively updating the POM according to the corpus changes instead. Besides increasing efficiency in this way, it also allows a user to trace the evolution of the ontology with respect to the changes in the underlying corpus.

Ontology Learning Tools from Social Network Data

Traditional Semantic Web deals with ontologies constructed mainly from text documents. Special ontology learning techniques deal almost exclusively with the problem of extracting and modeling the knowledge from text documents. The reason for this is that text is the most natural way of encoding information with the attached semantics. But text is not the only data modality which could be modeled using ontological structures. Ontological models can also be built from social network data. An example of these ontology learning tools is OntoGen (Fortuna et al., 2006).

OntoGen (Fortuna et al., 2006) is an ontology learning tool that can handle data represented as a set of feature vectors describing properties of ontological instances. Using several machine learning techniques (most prominent being k-means clustering, latent semantic indexing support vector machines, and uncertainty sampling active learning) OntoGen helps the user to construct an ontological structure directly from the data by providing suggestions and analyzing the user's decisions.

Ontology Learning Tools from Semi-structured Data

With the success of new standards for document publishing on the Web, there will be a proliferation of semi-structured data and formal descriptions of semi-structured data freely and widely available. HTML data, XML data, XML Document Type

Definitions (DTDs), XML-Schemata, and their likes add – more or less expressive – semantic information to documents. Ontologies can play a major role for allowing semantic access to these vast resources of semi-structured data. Though only few approaches do yet exist, extracting ontologies from these data and data descriptions may considerably leverage the application of ontologies and, thus, facilitate the access to these data. An example of this category is OntoBuilder (Modica et al., 2001).

The ***OntoBuilder*** (Modica et al., 2001) tool helps users in creating an ontology using as source semi-structured data coded in XML or in HTML. The modular architecture of the system consists of three main modules: the user interaction module, the observer module, and the ontology modeling module. The process followed to build an ontology has two phases: the training and the adaptation phase. In the training phase, an initial domain ontology is built using the data provided by the user. The adaptation phase aims in refining and generalizing the initial ontology. The user suggests browsing other web sites that contain relevant information for the domain. From each site, a candidate ontology is extracted and merged with the existing ontology. To perform this activity, a thesaurus can be used.

Ontology Learning Tools from Database Schemata

Ontologies play a key role in creating machine-processable Web content in order to promote Semantic Web. Extracting domain knowledge from database schemata can profitably support ontology development, as the Entity-Relationship (ER) model is an industrial standard for conceptually modeling databases.

In Xu et al. (2004) a formal approach is presented, as part of an automated tool for translating ER schemata into Web ontologies in the OWL Web Ontology Language. The tool can firstly read in an XML-codec ER schema produced with ER CASE tools such as PowerDesigner. Following the predefined knowledge-preserving mapping rules from ER schema to OWL DL (a sublanguage of OWL) ontology, it then automatically translates the schema into the ontology in both the abstract syntax and the RDF/XML syntax for OWL DL.

6.2.1.2 Ontologies Combination Tools

When one wants to reuse different ontologies together, those ontologies have to be *combined* in some way. This can be done by *integrating* the ontologies, which means that they are *merged* into a new ontology; alternatively, the ontologies can be *mapped*, that is, they can be kept separate. In both cases, the ontologies have to be *aligned*, which means that they have to be brought into mutual agreement. The alignment of concepts between ontologies is difficult, because it requires understanding of the meaning of concepts. Ontology alignment is concerned with the discovery of correspondences between ontologies. Ontology mapping is mostly concerned with the representation of these correspondences. Ontology merging is concerned with creating the union of ontologies, based on these correspondences. Aligning two ontologies, implies changes to at least one of them. Changes to an ontology will result in a new *version* of an ontology. If the ontologies are not represented in the same language, a *translation* is often required. It is not possible to make a strict distinction between the tools used for each ontology combination problem, because each tool usually provides support for several types of combination.

Ontologies Merging Tools

Ontologies merging is the creation of an ontology from two or more source ontologies. The new ontology will unify and in general replace the original source ontologies. There are two distinct approaches in ontologies merging. In the first approach, the input of the merging process is a collection of ontologies and the outcome is a new, merged ontology, which captures the original ontologies. In the second approach, the original ontologies are not replaced, but rather a "view", called *bridge ontology*, is created, which imports the original ontologies and specifies the correspondences using *bridge axioms*. OntoMerge (Dou et al., 2002), PROMPT (Noy and Musen, 2000), FCA-Merge (Stumme and Maedche, 2001), HCONE-merge (Kotis et al., 2006), and IF-Map (Kalfoglou and Schorlemmer, 2003) are tools that support the merging process.

OntoMerge (Dou et al., 2002) facilitates the creation of a "bridge" ontology, which imports the original ontologies and relates the concepts in these ontologies using a number of bridge axioms. It is an approach, in which the source ontologies are maintained after the merge operation. The output of the merge operation is not a completly merged ontology, but a bridge ontology which imports the source ontologies and has a number of Bridging Axioms, which are the translation rules used to connect the overlapping part of the source ontologies. It accepts a set of concepts or instance data based on one or more DAML ontologies, and a target ontology and produces the concepts or instance data translated to the target ontology.

PROMPT (Noy and Musen, 2000) is an algorithm and an interactive tool for the merging of two ontologies. It identifies a number of ontologies merging operations (merge classes, merge slots, merge bindings between a slot and a class, etc.) and a number of possible conflicts introduced by the application of these operations (name conflicts, dangling references, redundancy in the class hierarchy, and slot-value restrictions that violate class inheritance).

Ontologies Alignment Tools

Ontologies alignment is the process of discovering similarities between two source ontologies. It is generally described as the application of the so-called *Match* operator (Rahm and Bernstein, 2001). The input of the operator is a number of ontologies and the output is a specification of the correspondences between the ontologies. There are many different algorithms which implement the match operator. These algorithms can be generally classified along two dimensions. On the one hand, there is the distinction between schema-based and instance-based matching. A schema-based matcher takes different aspects of the concepts and relations in the ontologies and uses some similarity measure to determine correspondences (Noy and Musen, 2000). An instance-based matcher takes the instances which belong to the concepts in the different ontologies and compares these to discover similarities between the concepts (Doan et al., 2004). On the other hand, there is the distinction between element-level and structure-level matching. An element-level matcher compares properties of the particular concept or relation, such as the name, and uses these to find similarities (Noy and Musen, 2000). A structure-level matcher compares the structure (e.g., the concept hierarchy) of the ontologies to find similarities (Giunchiglia and Shvaiko, 2004). These matchers can also be combined (Ehrig and Staab, 2004). Falcon-AO (Jian et al., 2005), OLA (Euzenat et al., 2004), and COMA++ (Aumueller et al., 2005) can be considered as ontologies alignment tools.

Falcon-AO (Jian et al., 2005) is an automatic tool for aligning ontologies. There are two matchers integrated in Falcon AO. One is a matcher based on linguistic matching for ontologies, called LMO. The other is a matcher based on graph-matching for ontologies, called GMO. In Falcon-AO, GMO takes the alignments generated by LMO as external input and outputs additional alignments. Reliable alignments are gained through LMO as well as GMO. Reliability is obtained by observing the linguistic comparability and structural comparability of the two ontologies being compared. Falcon-AO (version 0.6) copes not only with ontologies of moderate size, but also with very large-scale ontologies. It integrates three distinguishing elementary matchers, to manage different alignment applications, and the integration strategy is totally automatic.

OLA (Euzenat et al., 2004) is another alignment tool that follows the similarity-based paradigm. It is dedicated to the alignment of ontologies expressed in OWL, with an emphasis on its restricted dialect of OWL, called OWL-Lite. More than a simple tool for automated alignment construction, OLA is designed as an environment for manipulating alignments. Indeed, the system offers the following services: parsing and visualization of (pairs of) ontologies; automated computation of similarities between entities from different ontologies; automated extraction of alignments from a pair of ontologies; manual construction of alignments; initialization of automated alignment construction by an existing alignment; visualization of alignments; comparison of alignments.

Ontologies Mapping Tools

Ontologies mapping is an important step to achieve knowledge sharing and semantic integration in an environment in which knowledge and information have been represented with different underlying ontologies. The process of ontologies mapping is a (declarative) specification of the semantic overlap between two ontologies. Given two ontologies A and B, mapping one ontology with another, means that for each concept (node) in ontology A, we try to find a corresponding concept (node), which has same or similar semantics, in ontology B and vice versa (Castano et al., 2007). The three main phases for any mapping process are: mapping discovery; mapping representation; mapping execution. CROSI Mapping System (Kalfoglou and Hu, 2005), GLUE (Doan et al., 2004), MAFRA (Maedche et al., 2002), OntoMap® (Schnurr and Angele, 2005), and H-Match (Castano et al., 2006) are such tools.

GLUE (Doan et al., 2004) is a system which employs machine-learning technologies to semi-automatically create mappings between heterogeneous ontologies based on instance data, where an ontology is seen as a taxonomy of concepts. It focuses on finding 1-to-1 mappings between concepts in taxonomies, although the authors say that extending matching to relations and attributes and involving more complex mappings (such as 1-to-n and n-to-1 mappings) is the subject of ongoing research.

MAFRA (Mapping FRAmework for distributed ontologies; Maedche et al., 2002) supports an interactive, incremental, and dynamic ontologies mapping process, the final purpose of which is to support ontology instance transformation. It adopts an open architecture in which concept mappings are realized through semantic bridges. A semantic bridge is a module that transforms source ontology instances into target ontology instances. The MAFRA toolkit supports a graphical user interface that provides domain experts with functionalities that are needed for the specification of semantic bridges. It has been implemented as a plug-in of KAON.

OntoMap® (Schnurr and Angele, 2005) is a plug-in for the ontology-management platform OntoStudio® that supports the creation and management of ontologies mappings. Mappings can be specified using a graphical representation, of the schema-view of the respective ontologies. It supports a number of elementary mapping patterns: concept to concept mappings, attribute to attribute mappings, relation to relation mappings, and attribute to concept mappings.

Ontologies Versioning Tools

As changes to ontologies are inevitable, it becomes very important to keep track of these changes, that is, the relation between successive revisions of one ontology and the relation between the ontology and its dependencies: instance data that conforms to the ontology; other ontologies that are built from, or import the ontology; applications that use the ontology. Therefore, a versioning tool is needed to handle revisions of ontologies and the impact on existing sources. In some sense, the versioning problem can also be regarded as a derivation of ontologies combination; it results from changes to individual ontologies. Although the problem is introduced by subsequent changes to a specific ontology, the most important problems are caused by the dependencies on that ontology. Ontologies versioning tools, such as SHOE (Heflin and Hendler, 2000), PROMPTDiff (Noy and Musen, 2002), OntoView (Klein and Fensel, 2001), Ontology Versioning Tool (De Leenheer et al., 2006), and SemVersion (Volkel and Groza, 2006) are about comparing ontology versions rather than about comparing independently developed ontologies.

OntoView (Klein and Fensel, 2001) is a system that helps ontology engineers to specify relations between ontology versions, in such a way, that interoperability between the versions of an ontology is improved. To provide a transparent interface to arbitrary versions of ontologies, OntoView keeps track of the conceptual relations and transformations between components of the ontology among different versions.

Such support is essential when ontologies are used on the Web and also useful for the collaborative development of ontologies.

Ontology Translation/Transformation Tools

Ontology transformation consists of transcribing an ontology from one form to another. This can include its expression in a different ontology language, or a reformulation in a restricted of a language (e.g., expressing automatically some non necessary OWL-Full ontology into OWL-DL), or with regard to a different vocabulary. Ontology transformation is useful for solving heterogeneity problems, when one wants to take advantage, in a particular context, of an ontology that has been developed in another context (i.e., using a different language). Ontology transformation is supported by a variety of tools. Some of them can be mere lexical or even be syntactic translators, but most will require the power of processing the ontology (i.e., inferring) in order to transform it. It is thus necessary to use the right tool: transformation systems are not version managers.

The term ontology translation is used in the literature to describe two different things. Under one understanding, ontology translation refers to the process of changing the formal representation of the ontology from one language to another (say from OWL to RDF or from Ontolingua to Prolog). This changes the syntactic form of the axioms, but not the vocabulary of the ontology. Works related to ontology translation under this understanding, leave the vocabulary of the ontology unaffected, dealing with the ontological axioms only. Under the second understanding, ontology translation refers to a translation of the vocabulary, in a manner similar to that of ontologies mapping. The difference between ontologies mapping and ontology translation is that the former specifies the function(s) that relate the two ontologies' vocabularies, while the latter applies this (these) function(s) to actually implement the mapping. OntoMorph (Chalupsky, 2000), WebODE translation system (Corcho, 2004), Transmorpher (Euzenat and Tardif, 2002), and RDFT (Omelayenko, 2002) are dedicated in ontology transformation.

OntoMorph (Chalupsky, 2000) is a system that supports the syntactic transformation of ontologies, using a language not very different from XSLT. It is however integrated with a knowledge representation system, which provides the ability to have semantically-grounded rules in the transformations. It uses syntactic rewriting via pattern-directed rewrite rules that allow the concise specification of sentence-level transformations based on pattern matching, and semantic rewriting. Syntactic rewriting is modulated via (partial) semantic models and logical inference.

6.2.1.3 Ontology Management Tools

Software tools are available to support the phases of ontology development. While ontology editors are useful during each step of the ontology building process, other types of ontology engineering tools are also needed along the way. Development projects often end up using numerous ontologies from external sources as well as existing and newly developed in-house ontologies. Ontologies from any source may progress through a series of versions. As a consequence, the careful management of this collection of heterogeneous ontologies becomes necessary so as to keep track of them. Some tools also support ontology mapping and linking, comparison, reconciliation and validation, merging, and converting into other forms. Other tools can help acquire, organize, and visualize the domain knowledge before and during the building of a formal ontology. Finally, other tools evaluate the existing ontologies, or use ontologies to annotate web documents, or support storing and querying ontologies. All these ontology management tools can be classified in four major categories that will be described in this section.

Ontology Evaluation Tools

Given the ever-increasing number of ontologies, there is an urgent need for evaluation, checking and validation tools. Ontology evaluation is the problem of assessing a given ontology from the point of view of a particular criterion of application, in order to determine if it would suit a particular purpose. It is a critical task, even more so when the ontology is the output of an automatic system, rather than the result of a conceptualization effort produced by a team of domain specialists and knowledge engineers. Ontology evaluation tools are responsible for checking ontology validation and consistency. These tools do not only validate the conformity to a standard of the syntactic representation in a special language, but also validate the semantic properties through the use of inference rules and other similar mechanisms.

Most evaluation approaches fall into four major categories: those based on comparing the released ontology to a predefined “golden standard”, which may be another hand-crafted ontology; those based on using the ontology in an integrated system and evaluating the performance of this system; those involving comparisons with a data source relative to the domain that is to be covered by the ontology and finally, those where the evaluation is performed purely by humans. Human assessment is performed by domain experts who try to assess how well the ontology meets a set of predefined criteria, requirements, standards, etc. Moreover, irrespective of the approach that is to be followed, an ontology can be evaluated at different layers. This is usually desirable, since an ontology is, in general, a complex structure and in most cases it is better to focus the evaluation on a particular aspect of the ontology.

Evaluation tools that evaluate the output ontology in a (semi)automatic way are needed, since the most widely used evaluation techniques are human-based and, consequently, may be biased. There also exist tools that evaluate the tools that develop ontologies.

Ontology Evaluation Tools for Tool Evaluation

Tools are evaluated for their technological properties, such as interoperability, performance, memory allocation, scalability, interface. An example of this category is OntoGenerator (Handschuh et al., 2001).

OntoGenerator (Handschuh et al., 2001) is an OntoEdit plug-in, focused on evaluating ontology tools’ performance and scalability.

Ontology Evaluation Tools for Ontology Evaluation

Ontologies are evaluated for their "language conformity" (i.e., to the syntax of a representation language) and for their "consistency" (i.e., to what extend they ensure consistency of specifications with respect to their semantics). OntoAnalyser (Handschuh et al., 2001), OntoClean (Guarino and Welty, 2002), AEON (Volker et al., 2005), ONE-T (Bouillon et al., 2002), and CleanONTO (Sleeman and Reul, 2006) are focused on ontology evaluation.

OntoAnalyser (Handschuh et al., 2001) is realized as a plug-in of OntoEdit. It focuses on evaluating ontology properties, particularly language conformity and consistency.

Ontology Storage and Querying Tools

Ontologies offer more services than simply representing knowledge and information. Human information consumers and web agents use and query ontologies and the resources committed to them. Since ontologies do not use traditional techniques and languages to represent information, the necessity to support adaptive storage and querying arises. However, the context of storing and querying knowledge has changed due to the wide acceptance and use of the Web as a platform for communicating knowledge. New languages for querying (meta)data based on web standards (e.g., XML, RDF, Topic Maps) have emerged, to enable the acquisition of knowledge from dispersed information sources, while the traditional database storage techniques have been adapted to deal with the peculiarities of the (semi)structured data on the web. So, these tools must provide full storage and query support to web-based ontology or metadata standards, such as RDF, RDFS, Topic Maps, DAML+OIL, and OWL. The tools presented here such as ICS-FORTH RDFSuite (Alexaki et al., 2001), Sesame (Broekstra et al., 2002), Redland (http://www.redland.opensource.ac.uk/demo/), Jena (McBride, 2001), RACER (http://www.sts.tu-harburg.de/∼r.f.moeller/racer) are indicative of this tendency. We can classify these tools, according to the language that they are based on:

Ontology Storage and Querying Tools Based on OWL

Jena (McBride, 2001) is a Java framework that provides a programming environment for RDF, RDFS and OWL, including a rule-based inference engine. The Jena2 release is more interesting for the ontology engineering process, as it provides an API that supports programmers who are working with ontology data based on RDF. This means that Jena2 offers features for handling most of ontology languages, namely XML, OWL, DAML+OIL, RDFS. It also provides generic RDF Resource and Property classes to model more directly the class and property expressions found in ontologies using the above languages, and the relationships between these classes and properties. Due to its storage abstraction, Jena enables new storage subsystems to be integrated. In fact, Jena provides statement-centric methods for manipulating an RDF model as a set of RDF triples, resource-centric methods for manipulating an RDF model as a set of resources with properties, as well as built-in support for RDF containers.

Ontology Storage and Querying Tools Based on DAML+OIL

TRIPLE (Sintek and Decker, 2001) is an inference and querying engine. It constitutes the implementation of the TRIPLE query language. It also contains a standalone DAML+OIL implementation with the following features: it parses DAML+OIL ontologies with Jena, provides output in various syntaxes, supports an external DL classifier that can be automatically invoked and its output can be returned in various formats.

Ontology Storage and Querying Tools Based on Topic Maps

Empolis K42 Knowledge Server constitutes a collaborative, web-based integrated authoring environment for capturing, expressing and delivering knowledge, which is able to import, export and merge Topic Maps (Magkanaraki et al., 2002).

Ontology Storage and Querying Tools Based on RDF Schema

Sesame (Broekstra et al., 2002) is an RDF Schema-based Repository and querying facility. It is a system consisting of a repository, a query engine and an administration module for adding and deleting RDF data and Schema information.

Ontology Storage and Querying Tools Based on RDF

ICS-FORTH RDFSuite (Alexaki et al., 2001) is a suite of tools for RDF metadata management, supporting RDF metadata processing for large-scale Web-based applications. It consists of tools for parsing, validating, storing and querying RDF descriptions.

Ontology-Based Annotation Tools

Ontology-based annotation refers to the process of creating metadata using ontologies as their vocabularies. Metadata is usually defined as "data about data", which aims at expressing the "semantics" of information. It is used to describe documents and applications, in order to improve information seeking and retrieval and its understanding and use. Metadata can be expressed in a wide variety of vocabularies and languages, and can be created and maintained with a variety of tools. Ontology-based annotation tools are primarily designed to allow inserting and maintaining ontology based markups in Web pages. Most of these tools such as OntoMat Annotizer (Handschuh et al., 2001), SHOE Knowledge Annotator (Heflin and Hendler, 2000), GATE (Cunningham et al., 2002), Melita (Ciravegna et al., 2002), AeroDAML (Kogut and Holmes, 2001), Amilcare (Ciravegna and Wilks, 2003), MnM (Vargas-Vera et al., 2002), S-Cream (et al., 2002), Magpie (Domingue et al., 2004), PANKOW system (Ciniamo et al., 2004), SemTag system (Dill et al., 2003), Photocopain (Tuffield et al., 2006), KIM system (Kiryakov et al., 2005), AKTive Media system (Chakravarthy et al., 2006), OntoELAN (Chebotko et al., 2004), and NOMOS (Niekrasz and Gruenstein, 2006) have appeared recently with the emergence of the Semantic Web. They use Information Extraction (IE) and Machine Learning (ML) techniques to propose semi-automatic annotations for Web documents.

Tools for Multimedia Documents' Annotation

Many language data are collected as audio and video recordings, which imposes a challenge to document indexing and retrieval. Annotation of multimedia data provides an opportunity for making the semantics of these data explicit and facilitates the searching for multimedia documents. So, a class of ontology-based annotation tools such as OntoELAN (Chebotko et al., 2004), AKTive Media system (Chakravarthy et al., 2006), Photocopain (Tuffield et al., 2006) are used as the ontology-based annotation tools for multimedia documents.

OntoELAN (Chebotko et al., 2004) is the first audio/video ontology-based annotation. It is an ontology-based linguistic annotator that features: support for loading and displaying ontologies specified in OWL; display a speech and/or video signals, together with their annotations; time linking of annotations to media streams; creation of a language profile, which allows a user to choose a subset of terms from an ontology and conveniently rename them if needed; creation of ontological tiers, which can be annotated with profile terms and, therefore, corresponding ontological terms; and saving annotations in XML format as multimedia ontology class instances and, linked with them, class instances of other ontologies used in ontological tiers.

Ontology-Based IE Tools

These tools perform semantic annotation with respect to an ontology without manual intervention (Bontcheva et al., 2006). Some examples of this category are PANKOW system (Ciniamo et al., 2004), KIM system (Kiryakov et al., 2005) and Magpie (Domingue et al., 2004).

Magpie (Domingue et al., 2004) is a suite of tools which supports the interpretation of web-pages and "collaborative sense-making". It annotates web-pages with metadata in a fully automatic fashion without manual intervention, by matching the text against instances in the ontology. It automatically creates a semantic layer for web pages using a user-selected ontology. Semantic layers are annotations of a web page, with a set of applicable semantic services attached to the annotated items. It can automatically track concepts found during a browsing session using a semantic log. The log allows trigger services to be activated when a specific pattern of concepts has been found. The same log can be used as a conceptual representation of the user's browsing history. Since all Magpie abilities are underpinned by ontological reasoning, this enables the users to use the history semantically rather than as a purely linear and temporal record of their activities.

Applying "Traditional" IE Tools

These tools do not incorporate ontologies into the system, but either use ontologies as a bridge between the IE system and the final annotation, as with AeroDAML (Kogut and Holmes, 2001), or rely on the user to provide the relevant information through manual annotation, as with Amilcare (Ciravegna and Wilks, 2003).

AeroDAML (Kogut and Holmes, 2001) is an annotation tool which applies IE techniques to automatically generate DAML annotations from web pages. It links most proper nouns and common types of relations with classes and properties in a DAML ontology. It consists of the AeroText IE system, together with components for DAML generation. A default ontology, which directly correlates to the linguistic knowledge base used by the extraction process, is used to translate the extraction results into a corresponding RDF model that uses the DAML+OIL syntax. This RDF model is then serialized to produce the final DAML annotation.

Ontology Visualization Tools

Ontologies are rich vocabularies, which implicitly contain more information than can be explicitly found in their text representation. Implicit information, such as the underlying structure of a data model, or which instances are most closely connected, is all contained in a graph. This information, though, is difficult, if not impossible, to extract from a text-based reading of the data. Tools that support the ontology visualization process are OntoViz (http://protege.cim3.net/cgi-bin/wiki.pl?OntoViz), and WSMOViz (Kerrigan, 2006).

WSMOViz (Kerrigan, 2006) is an integrated ontology engineering and visualization tool for WSMO (Feier and Domingue, 2005). This tool does not only allow the user to view WSML ontologies in a very clear way, but also to edit the ontology in the visual mode.

6.2.2 Integrated Ontology Engineering Environments

Ontology engineering, in addition to ontology construction, also supports mapping, management, maintenance and evolution of ontologies. There exist integrated collections of specialized tools that can support (fully or partially) more than one activities/processes of the ontology engineering life-cycle; these are called integrated ontology engineering environments. The need for these tools arises from the fact that the life cycle of ontology engineering is highly affected if the ontology is to be reused for building another ontology and vice-versa: different activities during the development of a specific ontology will be carried out if it is based on other ontologies that have already been built, or are under construction. This leads to interdependencies between the life cycles of ontologies. This interrelation between life cycles of several ontologies means that integration has to be approached globally (Fernandez-Lopez et al., 2000).

The integrated ontology engineering environments can represent all or some of the processes in the different phases of the ontology life cycle, such as the ontology generation (building, learning), ontology integration (merging, mapping, versioning, translating), and ontology management (evaluating, annotating, storing, querying, visualizing). Integrated ontology engineering environments such as KAON (Bozsak et al., 2002), Protégé (Noy et al., 2000), WebODE (Aprirez et al., 2001), OntoEdit (Sure et al., 2002), SWOOP (Kalyanpur et al., 2005), HCONE (Kotis and Vouros, 2003), SOBOLEO (Zacharias and Braun, 2007), ORIENT (http://apex.sjtu.edu.cn/projects/orient/), NeOn Toolkit (http://www.neon-toolkit.org), TopBraid Composer (http://www.topquadrant.com/topbraidvomposer.html), OBO-Edit (http://oboedit.org), and DIP Ontology Management Suite (SALERO Consortium, 2006) can be classified in being web-based or not, in supporting collaborative ontology engineering or not, etc.

SOBOLEO (Zacharias and Braun, 2007) is the acronym for SOcial BOokmarking and Lightweight Engineering of Ontologies. The system's goal is to support people working in some domain in the collaborative development of a shared vocabulary and a shared index of relevant web resources. With SOBOLEO it is possible to create, extend and maintain taxonomies according to the SKOS Core Vocabulary in a simple way. It also supports the collection and sharing of relevant web resources (bookmarks). These bookmarks can be annotated with concepts from the SKOS taxonomy or arbitrary tags for better retrieval. One instance or installation of SOBOLEO is meant to be used by a community with interest in building a shared vocabulary and a web index. Within this one instance, users create and maintain collaboratively, one taxonomy and one shared index of web resources.

Protégé (Noy et al., 2000) was developed by Stanford Medical Informatics at the Stanford University School of Medicine. It was developed originally for use in the field of clinical medicine and the biomedical sciences, but now it is being used in many areas where the concepts can be modeled as a class hierarchy. It is a Java based open-source tool for editing and managing ontologies. It is the most widely used domain-independent, freely available, platform-independent technology for developing and managing terminologies, ontologies, and knowledge bases in a broad range of application domains. With Protégé it is easy to create classes and hierarchies, to declare properties for classes, create instances and introduce values, all these under an environment consisting in menus, buttons, dialog boxes and easy to use graphic representations. Its core functionality can be extended in many ways by creating plug-ins. There are over 90 Protégé plug-ins providing advanced capabilities such as reasoning and inference support and visualization of large ontologies. For example, PROMPTDiff (Noy and Musen, 2002) automatically calculates structural differences between ontology versions, identifying which concepts have changed between versions. It identifies both simple and complex changes and it presents the comparison results to the user in an intuitive way. Users then can accept or reject the changes between concepts and instances. Jambalaya is another Protégé plug-in that provides an extensible, flexible, and scalable visualization environment for exploring, navigating, and understanding ontologies. ONTO-H (Benjamins et al., 2004) is a tab plug-in for the Protégé ontology editor that allows the creation of annotations of RTF documents. OntoLT (Buitelaar et al., 2004) implements the definition of mapping rules, with which, classes and properties can be extracted automatically from linguistically annotated text collections. Through the use of such rules, linguistic knowledge (context words, morphological and syntactic structure, etc.) remains associated with the constructed ontology and may be used subsequently in its application and maintenance, e.g., in knowledge markup, ontology mapping, and ontology evolution. OWLViz is also a Protégé plug-in which can be used to visualize ontologies built using the Protégé OWL plug-in. The OWL plug-in enables the creation and maintenance of ontologies described in the OWL syntax. A description logic inference engine, called RACER, is frequently used together with the Protégé OWL environment, as it provides reasoning services such as consistency checking and automated classification of concepts. Protégé can also support ontology merging with the PROMPT plug-in. The Protégé platform supports Web-based ontology viewing and collaboration and it provides different back-end storage mechanisms.

DIP Ontology Management Suite (http://kmi.open.ac.uk/projects/dip/index.php#publications) is an integrated set of tools for the effective management of ontologies, designed especially for handling and using ontologies as the underlying data model for Semantic Web Services. The whole suite consists of six major components that are developed as Eclipse plug-ins: Browsing and Editing (standard browsing and editing functionality); Mapping and Merging (allow to map and/or merge multiple ontologies); Versioning (controls the history of an ontology); Reporting (a graphical user interface for creating different types of diagram reports); Repository (for persistent storage and retrieval of ontology relevant data); Representation and Data Framework (middle layer and central API that provides transparent access to the suite's components).

6.3 Selecting the Appropriate Ontology Engineering And Management Tool

The continuous development of ontology editors and other tools for managing ontologies is an indication of the growing need for effective and universal knowledge representation in domains like the Semantic Web, Ubiquitous Computing, etc. These tools implement different knowledge models with different underlying knowledge representation paradigms, manage large upper level and general ontologies, and range from standalone to web-based and ubiquitous computing applications. Thus evaluation and comparison of these tools is important to help users determine which tool is best suited for their task.

In the last few years many studies evaluating ontology engineering tools have been published. Some authors have proposed general frameworks for the evaluation of ontology tools, i.e. the work presented by Duineveld and colleagues (Duineveld et al., 1999), the deliverable 1.3 of the OntoWeb project (2002), the conclusions attained in the First International Workshop on Evaluation of Ontology-based Tools (Angele and Sure, 2002), and Lambrix and colleagues (Lambrix et al., 2003). Others have presented more focused evaluations using specific criteria: Stojanovic and Motik (Stojanovic and Motik, 2002) analyzed the ontology evolution requirements fulfilled by the tools; Sofia Pinto and colleagues (Sofia Pinto et al., 2002) evaluated the support provided by the tools in ontology reuse process; In the Second International Workshop on Evaluation of Ontology-based Tools (Corcho et al., 2003) the interoperability of the tools was evaluated; and Gomez-Perez and Suarez-Figueroa (Gomez-Perez and Suarez-Figueroa, 2004) evaluated the ability of the tools to detect taxonomic anomalies.

From all these studies, it's evident that, on one hand, there is no "one fits all" generic framework that can be used for comparing ontology engineering tools: different categories of tools require very different comparison frameworks. For example, ontology engineering tools (ontology building tools and ontology learning tools) can easily be compared since they all support similar tasks such as definition of concepts, instances, and relations in a domain. Ontology combination tools (ontology merging, alignment, mapping, versioning, translation tools) however are so different from one another that direct comparison may not be possible. They differ in the type of input they require (e.g., instance data or no instance data), the type of output they produce (e.g., one merged ontology, pairs of related terms, articulation rules), modes of interaction and so on. This diversity makes comparing the performance of ontology combination tools to one another largely meaningless. On the other hand, we can summarize certain criteria as the basis for the comparative evaluation of integrated ontology engineering environments, such as (http://www.ontoweb.org/download/deliverables/D21_Final-final.pdf):

– Extensibility: measures how adaptable these environments may be to future technological advances. It is crucial for preserving a full development evolution of integrated ontology engineering environments.
– Maturity: measures how integrated ontology engineering environments may handle development problems, and even reduce the number and intensity of the future problematic situations. This criterion comprises the ability to deal constructively with real environments, the capacity to adapt to change, the capacity to relate and combine with other integrated environments, and so on.
– Portability: the ability to adapt any integrated ontology engineering environment, technique or method within a new environment without redeveloping it.
– Interoperability: the ability of systems to operate in conjunction with each other, encompassing communication protocols, hardware, software applications, and data compatibility layers.
– Ease-of-use: covers ease-of-learning, intuitiveness, efficiency and functionality. Simultaneously measures how long it takes for one to learn to use a certain product, how intuitive the product is, and how logical it is to use, create or modify a program.

To be able to decide what ontology engineering tools are needed for fulfilling actual and future requirements of an application, one needs to objectively evaluate existing tools. For several reasons, this is a difficult task. Firstly, for an evaluation to be unbiased, it must be designed and carried out by someone other than tool developers themselves. Otherwise, the evaluation setup and comparison parameters are inevitably skewed (often subconsciously). Secondly, it is often hard to come up with benchmark tasks and gold standards, because no two tools have been designed for the same purpose; thus, any attempt to evaluate a tool when performing restricted benchmark tasks often puts it in uses for which it was not designed. Thirdly, many of the criteria are, by their very nature, subjective. For example, when evaluating the quality of an ontology, we often don't have a single correct answer for how certain concepts should be represented or when evaluating the quality of ontology alignment, we often cannot agree on the precise relationships between concepts in source ontologies (http://co4.inriaples.fr/align/contest).

A systematic approach for comparing and selecting ontology engineering tools could be based on the taxonomy proposed in Fig. 6.1. Of course, this taxonomy should be enhanced with descriptions of specific properties of each class/tool describing various features and attributes of the class/tool, such as its architecture, the methodology that the tool follows, etc., as well as the allowed values for these properties. Then, available tools can be added as instances of specific classes (i.e., Protégé is an instance of the Integrated Ontology Engineering Environments class). In this classification, we have allocated each tool to one and only one class, but multiple inheritance should not be excluded, especially if we provide more abstract property descriptions.

This classification could be turned into an ontology of ontology engineering services, if we decompose tools into the functions each of them supports and then analyze the set of functions to form function classes. This ontology could then be questioned about what is the best ontology engineering tool for a specific application, in terms of the functions it supports, or which tools should be used to support the entire lifecycle of ontology engineering.

6.4 Conclusion

The next generation of semantic applications will be characterized by a large number of networked ontologies, some of them constantly evolving, most of them being locally, but not globally, consistent. In such scenarios, it is more or less infeasible to adopt current ontology management models, where the expectation is to have a single, globally consistent ontology, which serves the application needs of developers and possibly integrates a number of pre-existing ontologies. What is needed is a clear analysis of the complex relationships between ontologies in such networks, resulting in a formal model of networked ontologies that supports their evolution and provides the basis for guaranteeing their (partial) consistency in case one of the networked ontologies is changing. Open issues that are involved are among others ensuring consistency, evolution of ontologies and metadata, and reasoning. Developing tools that are able to meet these challenges is an essential requirement towards devising an ontology and metadata infrastructure that is powerful enough to support the realization of applications that are characterized by an open, decentralized, and ever changing environment.

Since ontologies encode a view of a given domain that is common to a set of individuals or groups in certain settings for specific purposes, mechanisms to tailor ontologies to the need of a particular user in his working context are required. The efficient dealing with a user's context posts several research challenges, such as formal representation of context, context based reasoning and context mapping. A promising application area of contextual information is user profiling and personalization. Furthermore, with the use of mobile devices and current research on ubiquitous computing, the topic of context awareness is a major issue for future IT applications. Intelligent solutions are needed to exploit context information and rapidly changing environments and unsteady information sources. Advanced tools for assigning context to a situation have to be developed, which pave the way to introduce ontology-based mechanisms into context-aware applications.

An increasing number of application scenarios depend on the integration of information from various sources that comes in different formats and is characterized by different formalization levels. For example, in many large engineering companies, information can be typically found in text documents, e-mails, graphical engineering documents, images, videos, sensor data, and so on, that is, information is stored in so-called cross-media resources. Taking this situation into account, the next generation of semantic applications will have to address various challenges in order to come up with appropriate solutions, such as ontology learning and metadata generation, information integration and advanced ontology mapping. Whereas individual (non)logical approaches exist to address these aspects, one lacks a coherent framework to handle these challenges in an integrated way. Providing tools that still scale up, or designing the interaction with the users in such complex scenarios is still an open research issue.

References

Alexaki, S., V. Christophides, G. Karvounarakis, D. Plexoudakis, and K. Tolle. 2001. The ICS-FORTH RDFSuite: Managing voluminous RDF description bases. 2nd International Workshop on the Semantic Web (SemWb2001), in conjuction with Tenth International World Wide Web Conference (WWW10), Hong Kong, 1–13.

Alfonseca, E., and P. Rodriguez. 2002. Automatically generating hypermedia documents depending on user goals. Workshop on Document Compression and Synthesis in Adaptive Hypermedia Systems, Malaga, Spain.

Angele, J., and Y. Sure, eds. 2002. Evaluation of ontology-based tools. In Proceedings of the 1st International Workshop EON2002. Siguenza, Spain. CEUR-WS.

Aprirez, J.C., O. Corcho, M. Fernandez-Lopez, and A. Gomez-Perez. 2001. WebODE: A scalable workbench for ontological engineering. In Proceedings of the First International Conference on Knowledge Capture (K-CAP) 21–23 Oct 2001, Victoria, BC.

Aumueller, D., H. Do, S. Massmann, and E. Rahm. 2005. Schema and ontology matching with coma++. In SIGMOD 2005 – Software Demonstration, Baltimore, MD.

Benjamins, V.R., J. Contreras, M. Blazquez, J.M. Dodero, A. Garcia, E. Navas, F. Hernandez, and C. Wert. 2004. Cultural heritage and the semantic web. In The Semantic Web: Research and Applications, First European Semantic Web Consortium (ESWS2004), eds. C. Bussler, J. Davies, D. Fensel, and R. Studer, 433–444. Berlin: Springer.

Bontcheva, K., H. Cunningham, A. Kiryakov, and V. Tablan. 2006. Semantic annotation and human language technology. In *Semantic web technologies: Trends and research in ontology-based systems,* eds. J. Davies, R. Studer, and P. Warren. Chichester, West Sussex: Wiley.

Bouillon, Y., O. Corcho, M. Fernandez-Lopez, and A. Gomez-Perez. 2002. A survey on ontology tools. OntoWeb: Ontology-based information exchange for knowledge management and electronic commerce.

Bozsak, E., M. Ehrig, S. Handschuh, A. Hotho, A. Madche, B. Motik, D. Oberle, C. Schmitz, S. Staab, L. Stojanovic, N. Stojanovic, R. Studer, G. Stumme, Y. Sure, and J. Tane, et al. 2002. KAON – Towards a large scale semantic web. In Proceedings of the 3rd International Conference on E-Commerce and Web Technologies (EC-Web2002), eds. K.Bauknecht, A.M. Tjoa, and G. Quirchmayr, vol. 2455 of LNCS, 304–313. Aix-en-Provence, France: Springer.

Broekstra, J., A. Kampman, and F. van Harmelen. 2002. Sesame: A generic architecture for storing and querying RDG and RDF schema. In The 1st International Semantic Web Conference (ISWC2002), Sardinia, Italy.

Buitelaar, P., P. Cimiano, and B. Magnini. 2005. *Ontology learning from text: Methods, applications and evaluation. Frontiers in artificial intelligence and applications*. Amsterdam: IOS Press.

Buitelaar, P., D. Olejnik, and M. Sintek. 2004. A Protégé plug-in for ontology extraction from text based on linguistic analysis. In Proceedings of the European Semantic Web Symposium (ESWS2004), Heraklion, Greece.

Castano, S., A. Ferrara, and S. Montanelli. 2006. Matching ontologies in open networked systems: Techniques and applications. *Journal on Data Semantics* 5:25–63. Springer.

Castano, S., A. Ferrara, S. Montanelli, G.N. Hess, and S. Bruno. 2007. State of the art on ontology coordination and matching. D4.4. Bootstrapping Ontology Evolution with Multimedia Information Extraction.

Chakravarthy, A., F. Ciravegna, and V. Lanfranchi. 2006. Cross-media document annotation and enrichment. In Proceedings of the 1st semantic web authoring and annotation workshop (SAAW2006), Athens.

Chalupsky, H. 2000. OntoMorph: A translation system for symbolic knowledge. In Proceedings of 7th International Conference on Knowledge Representation and Reasoning (KR), Breckenridge, CO, 471–482.

Chebotko, A., L. Yu Deng, F. Fotouhi, A. Aristar, H. Brugman, A. Klassmann, H. Sloetjes, A. Russel, and P. Wittenburg. 2004. OntoELAN: An ontology-based linguistic multimedia annotator. In Proceedings of IEEE 6th International Symposium on Multimedia Software Engineering, Miami, 329–336.

Ciniamo, P., S. Handschuh, and S. Staab. 2004. Towards the self-annotating web. In Proceedings of WWW2004, New York, NY.

Ciniamo, P., and J. Volker. 2005. *Text2onto. In natural language processing and information (LNCS)*, 227–238. Berlin and Heidelberg: Springer.

Ciravegna, F., A. Dingli, D. Petrelli, and Y. Wilks. 2002. Timely and non-intrusive active document annotation via adaptive information extraction. In Semantic Authoring, Annotation and Knowledge Markup (SAAKM2002). ECAI, Lyon, France.

Ciravegna, F., and Y. Wilks. 2003. Designing adaptive information extraction for the semantic web in Amilcare. In *Annotation for the semantic web,* eds. S. Handschuh, and S. Staab. Amsterdam: IOS Press.

Corcho, O. 2004. A declarative approach to ontology translation with knowledge preservation. PhD thesis, Universidad Politecnica de Madrid.

Corcho. O., A. Gomez-Perez, D.J. Guerrero-Rodriguez, D. Perez-Rey, A. Ruiz-Cristina, T. Sastre-Toral, and M.C. Suarez-Figueroa. 2003. Evaluation experiment of ontology tools' interoperability with the WebODE ontology engineering workbench. In Proceedings of the 2nd International Workshop on Evaluation of Ontology-based Tools (EON2003), Sanibel Island, FL.

Cunningham. H., D. Maynard, V. Tablan, C. Ursu, and K. Bontcheva. 2002. Developing language processing components with GATE. Available at www.gate.ac.uk.

De Leenheer, P., J. Kopecky, and E. Sharf. 2006. Requirements and design for an ontology versioning tool. Data, Information and Process Integration with Semantic Web Services. D2.4.

Dill, S., N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingram, T. Kanungo, S. Rajagopalan, A. Tomkins, J.A. Tomlin, and J.Y. Zien. 2003. SemTag and seeker: Bootsrapping the semantic web via automated semantic annotation. In Proceedings of WWW2003, Budapest, Hungary.

Doan, A., J. Madhaven, P. Domingos, and A. Halevy. 2004. Ontology matching: A machine learning approach. In *Handbook on ontologies in information systems,* eds. S. Staab, and R. Studer, 397–416. Heildelberg: Springer.

Domingue, J., M. Dzbor, and E. Motta. 2004. Magpie: Supporting browsing and navigation on the semantic web. In Proceedings ACM Conference on Intelligent User Interfaces (IUI), eds. N. Nunes, and C. Rich, 191–197. New York, NY: ACM Press.

Dou, D., D. McDermott, and P. Qi. 2002. Ontology translation by ontology merging and automated reasoning. In Proceedings EKAW2002 Workshop on Ontologies for Multi-Agent Systems, Spain, 3–18.

Duineveld, A.J., R. Stoter, M.R. Weiden, B. Kenepa, and V.R. Benjamins. 1999. Wondertools? A comparative study of ontological engineering tools. In Proceedings of the 12th International Workshop on Knowledge Acquisition, Modeling and Management (KAW1999). Banff, Canada: Kluwer Academic Publishers.

Ehrig, M., and S. Staab. 2004. QOM-quick ontology mapping. In Proceedings of the Third International Semantic Web Conference (ISWC2004), LNCS, eds. F. van Harmelen, S. McIlraith, and D. Plexoudakis, 683–696. Springer: Heidelberg.

Euzenat, J., D. Loup, M. Touzani, and P. Valtchev. 2004. Ontology Alignment with OLA. In Proceedings of the 3rd EON Workshop, 3rd International Semantic Web Conference, Hiroshima.

Euzenat, J., and L. Tardif. 2002. Xml transformation flow processing. *Markup Languages: Theory and Practice* 3(3):28–311.

Feier, C., and J. Domingue. 2005. D3.1v0.12 WSMO Primer, WSMO Working Draft.

Fernandez-Lopez, M., A. Gomez-Perez, and M.D. Rojas Amaya. 2000. Ontology's crossed life cycles. In Proceedings of the 12th International Conference in Knowledge Engineering and Knowledge Management. Methods, models, and tools (EKAW2000), vol. 1937, 155–179, Juan-les-Pins, France: Springer.

Fortuna, B., M. Grobelnik, and D. Mladenic. 2006. System for semi-automatic ontology construction. In Proceedings of the 3rd European Semantic Web Conference ESWC2006, Budva, Montenegro.

Giunchiglia, F., and P. Shvaiko. 2004. Semantic matching. *The Knowledge Engineering Review* 18(3):265–280.

Gomez-Perez, A., and M.C. Suarez-Figueroa. 2004. Evaluation of RDF(S) and DAML+OIL import/export services within ontology platforms. In Proceedings of the 3rd Mexican International Conference on Artificial Intelligence, Mexico, 109–118.

Guarino, N., and C. Welty. 2002. An Overview of OntoClean. In *The handbook on ontology,* eds. S. Staab, and R. Studer, 151–172. Berlin: Springer.

Jian, N., W. Hu, G. Cheng, and Y. Qu. 2005. Falcon-AO: Aligning ontologies with Falcon. In Proceedings of the K-CAP Workshop on Integrating Ontologies, Banff, Canada 85–91.

Jones, S., and G.W. Paynter. 2002. Automatic extraction of document keyphrases for use in digital libraries: Evaluation and applications. *Journal of the American Society for Information Science and Technology (JASIST)* 53(8):653–677.

Handschuh, S., S. Staab, and F. Ciravegna. 2002. S-CREAM – Semi-automatic CREAtion of metadata. In 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW2002), Siguenza, Spain, 358–372.

Handschuh, S., S. Staab, and A. Maedche. 2001. CREAM – Creating relational metadata with a component-based, ontology-driven annotation framework. In First international conference on knowledge capture (KCAP2001), Victoria, Canada, eds. Y. Gil, M. Musen, and J. Shavlik, 76–83. New York: ACM Press. 1-581113-380-4.

Heflin, J.D., and J.A. Hendler. 2000. Searching the web with SHOE. In Artificial Intelligence for Web Search, AAAI Worksop, Menlo Park, CA, 35–40.

Kalfoglou, Y., and B. Hu. 2005. CMS: CROSI mapping system – Results of the 2005 ontology alignment contest. In Proceedings of the K-Cap'05 Integrating Ontologies Workshop, Canada, 77–84.

Kalfoglou, Y., and M. Schorlemmer. 2003. If-map: An ontology-mapping method based on information-flow theory. *Journal on Data Semantics* 1:98–127.

Kalyanpur, A., B. Parsia, E. Sirin, B. Cuenca-Grau, and J. Hendler. 2005. Swoop: A "web" ontology editing browser. *Journal of Web Semantics* 4(2): 144–153.

Kerrigan, M. 2006. WSMOViz: An ontology visualization approach for WSMO. In 10th International Conference on Information Visualization, London.

Kiryakov, A, B. Popov, I. Terziev, D. Manov, and D. Ognyanoff. 2005. Semantic annotation, indexing and retrieval. *Journal of Web Semantics* 2(1): 49–79.

Klein, M., and D. Fensel. 2001. Ontology versioning for the semantic web. In Proceedings of the 1st International Semantic Web Working Symposium, 75–91. Stanford, CA: Stanford University.

Kogut, P., and W. Holmes. 2001. AeroDAML: Applying Information Extraction to Generate DAML Annotations from Web Pages. In 1st International Conference on Knowledge Capture (K-CAP2001). Workshop on Knowledge Markup and Semantic Annotation, Victoria.

Kotis, K., and G. Vouros. 2003. Human-centered ontology management with HCONE. In Proceedings of Ontologies and Distributed Systems Workshop, IJCAI2003 Conference, Acapulco, Mexico.

Kotis, K., G.A. Vouros, and K. Stergiou. 2006. Towards automatic merging of domain ontologies: The HCONE-merge approach. *Journal of Web Semantics* 4(1):60–79.

Lambrix, P., M. Habbouche, and M. Perez. 2003. Evaluation of ontology development tools for bioinformatics. *Bioinformatics* 19(12):1564–1571.

Maedche, A., B. Motik, N. Silva, and R. Volz. 2002. Mafra-a mapping framework for distributed ontologies. In Proceedings of the 13th European Conference on Knowledge Engineering and Knowledge Management EKAW-2002, Madrid, Spain.

Maedche, A., and S. Staab. 2001. Ontology learning for the semantic web. *IEEE Intelligent Systems, Special Issue on the Semantic Web* 16(2):72–79.

Magkanaraki, A., G. Karvounarakis, T.T. Anh, V. Christophides, and D. Plexoudakis. 2002. Ontology storage and querying. Technical Report No.308. Foundation for Research and Technology Hellas, Institute of Computer Science, Information System Laboratory.

McBride, B. 2001. Jena: Implementing the RDF model and syntax specification. In Proceedings of the 2nd International Workshop on the Semantic Web (SemWeb2001), Hongkong.

Modica, G., A. Gal, and H.M. Jamil. 2001. The use of machine-generated ontologies in dynamic information seeking. In The Proceedings of the 6th International Conference on Cooperative Information Systems, Trento, Italy, LNCS. Heidelberg: Springer.

Niekrasz, J., and A. Gruenstein. 2006. NOMOS: A semantic web software framework for annotation of multimodal corpora. In Proceedings of the 5th Conference on Language Resources and Evaluation (LREC2006), Genova, Italy.

Noy, N., R. Fergerson, and M. Musen. 2000. *The knowledge model of Protégé-2000: Combining interoperability and flexibility of Lecture Notes in Artificial Intelligence (LNAI)*, vol. 1937, 17–32, Juan-les-Pins, France: Springer.

Noy, N.F., and M.A. Musen. 2000. PROMPT: Algorithm and tool for automated ontology merging and alignment. In Proceedings 17th National Conference on Artificial Intelligence (AAAI2000), Austin, TX.

Noy, N.F., and M.A. Musen. 2002. PromptDiff: A fixed-point algorithm for comparing ontology versions. In 18th National Conference on Artificial Intelligence (AAAI2002). Edmonton, AB.

Omelayenko, B. 2002. Integrating vocabularies: discovering and representing vocabulary maps. In Proceedings of 1st International Semantic Web Conference (ISWC2002), Chia Laguna (IT), 206–220.

OntoWeb Consortium: A survey of ontology tools. IST Project IST-2000-29243, OntoWeb: Ontology-based Information Exchange for Knowledge Management and Electronic Commerce, Deliverable1.3, 2002.

Rahm, E., and P.A. Bernstein. 2001. A survey of approaches to automatic schema matching. *VLDB Journal: Very Large Data Bases* 10(4):334–350.

SALERO Consortium. 2006. Representation techniques for multimedia objects. Deliverable 3.1.1.

Schnurr, H.P., and J. Angele. 2005. Do not use this gear with a switching lever! Automotive industry experience with semantic guides. In 4th International Semantic Web Conference (ISWC2005), 1029–1040. Berlin and Heidelberg: Springer.

Sintek, M., and S. Decker. 2001. TRIPLE-An RDF query, inference, and transformation language. In Proceedings of the Deductive Databases and Knowledge Management Workshop (DDLP2001), Japan.

Sleeman, D., and Q. Reul. 2006. CleanONTO: Evaluating taxonomic relationships in ontologies. In WWW2006, Edinburg.

Sofia Pinto, H., D.N. Peralta, and N.J. Mamede. 2002. Using Protege-2000 in reuse processes. In Proceedings of the International Workshop on Evaluation of Ontology-Based Tools. (EON2002), Siguenza, Spain.

Stojanovic, L., and B. Motik. 2002. Ontology evolution within ontology editors. In Proceedings of the International Workshop on Evaluation of Ontology-Based Tools. (EON2002), Siguenza, Spain.

Stumme, J., and A. Maedche. 2001. FCA-Merge: Bottom-up merging of ontologies. In 7th International Conference on Artificial Intelligence (IJCAI2001), Seattle, 225–230.

Sure, Y., M. Erdmann, J. Angele, S. Staab, R. Studer, and D. Wenke. 2002. OntoEdit: Collaborative ontology development for the semantic web. In Proceedings of the 1st International Semantic Web Conference: The Semantic Web (ISWC2002), Sardinia, Italy, eds. I. Horrocks, and J.A. Hendler, vol. 2342 of Lecture Notes in Computer Science (LNCS), 221–235. Heidelberg: Springer.

Tuffield, M.M., S. Harris, D.P. Dupplaw, A. Chakravarthy, C. Brewster, N. Gibbins, K. O'Hara, F. Ciravegna, D. Sleeman, N.R. Shadbolt, and Y. Wilks. 2006. Image annotation with photocopain. In Proceedings of Semantic Web Annotation of Multimedia (SWAMM2006) Workshop at the World Wide Web Conference, Edinburgh.

Vargas-Vera, M., E. Motta, J. Domingue, M. Lanzoni, A. Stutt, and F. Ciravegna. 2002. MnM: Ontology driven semi-automatic and automatic support for semantic markup. In 13th International Conference on Knowledge Engineering and Management (EKAW2002), eds. A. Gomez-Perez, V.R. Benjamins, 379–391. Heidelberg: Springer.

Velardi, P., R. Navigli, A. Cuchiarelli, F. Neri. 2005. Evaluation of ontolearn, a methodology for automatic population of domain ontologies. In *Ontology learning from text: Methods, applications and evaluation*, eds. P. Buitelaar, P. Ciniamo, and B. Magnini. Amsterdam: IOS Press.

Volkel, M., and T. Groza. 2006. SemVersion: An RDF-based ontology versioning system. In ICIW2006, Guadeloupe, France.

Volker, J., D. Vrandecic, and Y. Sure. 2005. Automatic evaluation of ontologies (AEON). In Proceedings of the 4th International Semantic Web Conference, Galway, Ireland.

Xu, Z., X. Cao, Y. Dong, and W. Su. 2004. Formal Approach and Automated Tool for Translating ER Schemata into OWL Ontologies. In *Advances in knowledge discovery and data mining LNC,* 464–475. Berlin and Heidelberg: Springer.

Zacharias, V., and S. Braun. 2007. SOBOLEO – Social Bookmarking and Leighweight Engineering of Ontologies. Workshop on Social and Collaborative Construction of Structured Knowledge (CKC), 16th International World Wide Web conference, (WWW2007), Banff, AB.