International Journal of Enhanced Research in Management & Computer Applications
ISSN: 2319-7471, Vol. 4 Issue 10, October-2015

A Comparative study of Ontology Languages and tools

Yatin Chopra[1], Gautam[2]

[1,2]M. Tech. Scholars, Dept. of CSE, MD University, Rohtak, Haryana

ABSTRACT

Aims: Ontologies Languages and Tools have been becoming a hot research topic for the application in artificial intelligence, semantic web, Software Engineering and information Architecture. Ontology is a formal representation of set of concepts within a domain and relationships between those concepts. It is used to reason about the properties of that domain and may be used to define the domain. An ontology language is a formal language used to encode the ontologies. A number of research languages have been designed and released during the past few years by the research community. They are both proprietary and standard based. In this paper a comparative study has been reported on the different features and issues of these languages and its tools.

Keywords: ontologies, languages, tools, semantic, computer applications.

INTRODUCTION

Ontologies, which are used in order to support interoperability and common understanding between the different parties, are a key component in solving the problem of semantic heterogeneity, thus enabling semantic interoperability between different web applications and services. The primary use of the word "ontology" is in the discipline of philosophy, where it means "the study or theory of the explanation of being"; it thus defines an entity or being and its relationship with and activity in its environment. In other disciplines, such as software engineering and AI, it is defined as "a formal explicit specification of a shared conceptualization". The foundations of this definition are:

- All knowledge (e.g. the type of concepts used and the constraints on their use) in ontology must have an explicit specification.
- An ontology is a conceptualisation, which means it has a universally comprehensible concept.
- "Shared" indicates an agreement about the meaning in such domains. In other words, an ontology should capture consensual knowledge accepted by the communities.
- "Formal" refers to the grounding of representation in well understood logic, and any ontology should be machine-processable.

Ontology engineering (or ontology building) is a subfield of knowledge engineering that studies the methods and methodologies for building ontologies. It studies the ontology development process, the ontology life cycle, the methods and methodologies for building ontologies, and the tool suites and languages that support them. Ontology engineering aims to make explicit the knowledge contained within software applications, and within enterprises and business procedures for a particular domain. Ontology engineering offers a direction towards solving the interoperability problems brought about by semantic obstacles, such as the obstacles related to the definitions of business terms and software classes. Ontology engineering is a set of tasks related to the development of ontologies for a particular domain [1].

Most ontologies describe individual (to instances), classes (concepts), attributes and relations. Common components of ontologies include individuals, classes, attributes, relating, functions terms, restriction, rules axioms and events. Ontologies are used in artificial intelligence, semantic web, software engineering, Biomedical information, library science and information architecture as a form of knowledge representation about the world or some part of it. An ontology language is a formal language used to encode the ontology.

There are a number of such languages for ontologies, both proprietory and standard-based such as common Algebraic specification language, common logic, CycL, DOGMA, Gellish, IDEF5, KIF, RIF, and OWL. Ontology languages can be classified as:

1) Logical Languages

- First order predicate logic
- Rule based logic
- Description logic

2) Frame based Languages

- Similar to relational databases

3) Graph based Languages

- Semantic network
- Analogy with the web is rationale for the semantic web

Even though ontologies have a long history in Artificial Intelligence (AI), the meaning of this concept still generates a lot of controversy in discussions, both within and outside of AI. We follow the classical AI definition: an ontology is a formal specification of a conceptualisation, that is, an abstract and simplified view of the world that we wish to represent, described in a language that is equipped with a formal semantics. In knowledge representation, an ontology is a description of the concepts and relationships in an application domain. Depending on the users of this ontology, such a description must be understandable by humans and/or by software agents. In many other field – such as in information systems and databases, and in software engineering – an ontology would be called a conceptual schema.

An ontology is formal, since its understanding should be non ambiguous, both from the syntactic and the semantic point of views. Researchers in AI were the first to develop ontologies with the purpose of facilitating automated knowledge sharing. Since the beginning of the 90's, ontologies have become a popular research topic, and several AI research communities, including knowledge engineering, knowledge acquisition, natural language processing, and knowledge representation, have investigated them. More recently, the notion of an ontology is becoming widespread in fields such as intelligent information integration, cooperative information systems, information retrieval, digital libraries, e-commerce, and knowledge management. Ontologies are widely regarded as one of the foundational technologies for the Semantic Web: when annotating web documents with machine-interpretable information concerning their content, the meaning of the terms used in such an annotation should be fixed in a (shared) ontology. Research in the Semantic Web has led to the standardisation of specific web ontology languages.

ONTOLOGY LANGUAGES

How do we describe a particular domain? Let us consider the domain of courses and lecturers at Griffith University. First we have to specify the "things" we want to talk about. Here we will make a first, fundamental distinction. On one hand we want to talk about particular lecturers, such as David Billington, and particular courses, such as Discrete Mathematics. But we also want to talk about courses, first year courses, lecturers, professors etc. What is the difference? In the first case we talk about individual objects (resources), in the second we talk about classes (also called concepts) which define types of objects.

A class can be thought of as a set of elements, called the extension of the class. Individual objects that belong to a class are referred to as instances of that class.

An important use of classes is to impose restrictions on what can be stated. In programming languages, typing is used to prevent nonsense from being written (such as A + 1, where A is an array; we lay down that the arguments of + must be numbers). The same is needed in RDF. After all, we would like to disallow statements such as:

– Discrete Mathematics is taught by Concrete Mathematics.
– Room MZH5760 is taught by David Billington.

Primitive ontology languages are Simple HTML Ontology Extension (SHOE), Ontobroker and Ontology Inference Layer (OIL) [2]. Several ontology languages have evolved and coordinated by different organizations, such W3C and US Department of Defense. Afterward, some other ontology languages that were designed in pursuit of enhancing and incorporating reasoning to ontology are RDF/RDFS, DAML, OIL, OWL, XTM and WSMO. RDF is a domain independent ontology language the makes no assumptions about a particular domain of use, and the user is required to define the needed terminology in a schema language called RDFS [3]. Hence RDF/RDFS are jointly used to define concepts of a given domain. DARPA sponsored DAML-ONT and OIL were later merged to form what was then referred to as DAML+OIL and were later submitted to World Wide Web (W3C) [4]. OWL is another ontology language that was designed and been coordinated by W3C. OWL was a bid to improve on the expressiveness of RDF/RDFS by the W3C.

OWL has three variants which are Web Ontology Language Full (OWL) Full, Web Ontology Language Lite (OWL Lite) and Web Ontology Language Defeasible Logic (OWL DL). OWL2 has OWL2-RL, OWL2- QL and OWL2-EL as the three variants it consists of. Another ontology language is XMT and was originally designed to handle the construction of indexes, glossaries, thesauri and table of contents. Lastly, WSMO is an ontology used for modelling services that are been searched [5]. A major characteristic of data repository is its provision for the manipulation and query of the underlying data. Quite a number of semantic web query languages have emerged are tailored to either a particular ontology language or related ontology languages. And some of these query languages include Sparql Protocol and RDF Query Language (SPARQL), Semantic Query Web Rule Language (SQWRL) and RDF Query Language (RQL). SPARQL queries are pattern matching queries on triples that constitute an RDF data graph [2]. SQWRL is a query language used in querying OWL and are built interoperability with a rule language called Semantic Web Rule language (SWRL) in mind. They take the antecedent of SWRL and effectively treat it as a pattern specification for a query [6] RDF Query Language (RQL) is a query language used to query RDF. It illustrates a number of features that will be part of any reasonable query language in RDF and RDFS, such as path expressions and schema awareness.

ONTOLOGY CONSTRUCTION

Provision to share and re-use knowledge is what is needed for the knowledge based system. For a large scale domain, ontology is very useful and has the ability to store the relations that are connected. The way the information is expressed needs to be changed as it is not an easy task to teach computers to comprehend natural languages. Ontology is a set of assertions and relations among the objects for specifying the concepts involved in the specific domains [9]. Ontology development enables [10]:

- Sharing common understanding of knowledge among people and machine
- Reuse of the domain and expert knowledge
- Making the domain assumptions explicit
- Increase interoperability among various domains
- Increase the scalability

There are multiple layered concepts in any knowledge system. Domain areas can often be organized in a tree structure composed of a super-concept, sub-conceptand their relevant relationships. . The tree structure for the knowledge of ontology-based search engine facilitates the search of web knowledge simply by adding structure to the largely unstructured web. In addition, the ontology can be indexed to further facilitate efficient searches. Reasoning on general concepts can be performed by matching various concepts in different expression and relevant classes [11]. Ontology is created from scratch by extracting information from domain experts and merging already existing ontology into new ontology. It forms the basis for syntactic and semantic metadata which can be used for annotating or tagging content [12]. Many of the currently available ontology supported languages are: XML, RDF and OWL. The available tools are: Protégé, Model Futures OWL Editor, TopBraid Suite, OntoLingua, OntoEdit, WebODE, KAON, ICOM, DOE, WebOnto, Medius Visual Ontology Modeler, LinKFactory Workbench, K-Infinity and OntoStudio. Some of these are available as freeware tools. In this paper, we have chosen, Protégé, SWOOP (open source tools) and OntoStudio, TopBraid (commercially available) for the analysis

Class hierarchies

Once we have classes we would also like to establish relationships between them. For example, suppose that we have classes for :

– staff members

– academic staff members

– professors

– associate professors

– assistant professors

– administrative staff members

– technical support staff members.

These classes are not unrelated to each other. For example, every professor is an academic staff member. We say that professor is a subclass of academic staff member, or equivalently, that academic staff member is a superclass of professor [15]. The subclass relationship is also called subsumption. The subclass relationship defines a hierarchy of classes. In general, A is a subclass of B if every instance of A is also an instance of B. A hierarchical organisation of classes has a very important practical significance, which we outline now. Consider the range restriction Courses must be taught by academic staff members only. Suppose Michael Maher was defined as a professor. Then, according to the restriction above, he is not allowed to teach courses. The reason is that there is no statement which specifies that Michael Maher is also an academic staff member. Obviously it would be highly counterintuitive to overcome this difficulty by adding that statement to our description. Instead we would like Michael Maher to inherit the ability to teach from the class of academic staff members [18].

ONTOLOGY TOOLS

Concise descriptions of each software tool are compiled and then reviewed by the organization currently providing the software for commercial, open, or restricted distribution. The descriptions are factored into a dozen different categories covering important functions and features of the software. These categories are summarizing the results.

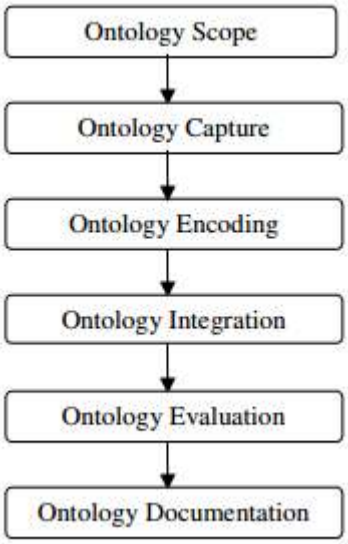

Fig.: Ontology Construction methodology

We have used only Four “popular and accepted” ontology authoring tools taking into consideration the advantages of these tools. Tools that provide support for the different phases of the ontology engineering process are referred to as ontology building tools. These tools are used for building a new ontology either from scratch or by reusing existing ontologies, which usually supports editing, browsing, documentation, export and import from different formats, views; libraries and they may have attached inference engines, etc. [20]. The ontology editors are tools that allow users to visually manipulate, inspect, browse and code ontologies and support in this way the ontology development and maintenance task [21]. In this section, we will provide a broad overview of some of the available ontology editor tools with a brief description of each tool, presenting the group that has developed it, its main features and functionalities, its URL etc.

Consider the task of building an ontology for a large state university system. Typically, multiple relatively autonomous groups (faculty, programs, departments, colleges) contribute parts of such an ontology that pertain to their domains of expertise or responsibility. The ontology for the university system should be a semantically coherent integration of the

constituent ontologies developed by the individual groups. Hence, there is a need for collaborative ontology construction tools. We enumerate below, some desiderata of such collaborative ontology construction tools.

Local Terminology: Terms used in different ontologies e.g., the department name, research topics, and graduate student status, etc. should be given unique identifiers. This is necessary to avoid name conflicts when merging two independently developed ontologies and to avoid unwanted interactions among modules. Manual processing of such name conflicts does not scale up with increase in size, number, and complexity of ontologies [22].

Localized Semantics: Collaborative ontology construction requires different groups to adapt or use of ontologies that were independently developed by other groups. However, unrestricted use of entities and relationships from different ontologies can result in serious semantic conflicts, especially when the ontologies in question represent local views of the world from the respective points of view of the ontology producers.

Ontology Evolution: Ontology construction is usually an iterative process. This is especially true in emerging areas of science in which there is little consensus concerning the basic entities and assumed relationships among entities (i.e., ontological commitments). A small change in one part of an ontology may be propagated in an unintended and hence undesirable manner across the entire ontology [23].

Distinction between Organizational and Semantic Hierarchies: an organizational hierarchy of ontology entities may be different in structure from the semantic hierarchy. Furthermore, there might be a need for an organizational hierarchy even when a semantic hierarchy is missing. For example, given

HistoryDeparment ¥AcademicDepartments

HistoryDeparmentHall ∈ Building

HistoryStudentClubs ¥ StudentOrganization

Suppose that it is now desired to state that the above three concepts are all about history department Introducing a new common super-class, say HistoryDepartmentRelated, instead of clarifying the semantics associated with the concepts in question, will introduce logical ambiguity.. This problem is even worse for properties because of the distinction between datatype property and object property in ontology languages such as OWL. It's usually hard to design a superproperty when both datatype extension and object type extension are possible in future.

Ontology Reuse: In collaborative design of ontologies, it often makes sense to reuse parts of existing ontologies. However, lack of modularity and localized semantics in ontologies forces an all or nothing choice with regard to reuse of an existing ontology. For example, a university library may want to reuse part of Congress Library Catalog ontology in creating its own ontology. Nevertheless, because ontology languages such as OWL do not support the importing and reuse of only partial existing ontology.

Knowledge Hiding: In many applications, the provider of an ontology may not wish, because of copyright considerations or privacy or security concerns, to make the entire ontology visible to the outside while willing to expose certain parts of the ontology to certain subsets of users [24].

A good choice of ontology developing software and editors helps developers in designing a more correct ontology for applications. Several ontology editors in cooperates other plugins that onerously help developer improve their ontology design, visualize it, and check for consistency of information modelled. It can be used to develop both simple and complex ontology-based applications. And since they also help in developing intelligent systems, they allow rule systems – a combination of rule engine and rules implemented with semantic web rule languages – so as to yield a wide range of intelligent systems. Protégé provides supports such as DIG reasoned, Pellet, check for consistency, check for taxonomy, computation of inferred types and ontology visualizer plug-in like OWLVIZ. It is open source software, with large online support groups who develop ontology for applications in e-commerce, biomedicine, organizational and institutional purposes. Protégé fully supports OWL 1 and the OWL 2 discussed in this paper, and RDF. Whereas other ontology editors have some limitations that disadvantage them in enjoying wide user community, Protégé continues to be in use by many developers. For instance, Ominigator is strictly used for creating XTM ontology, OntEdit, though still in use, but not as pronounced as Protégé. Furthermore, we noted that OWL is one of the most powerful and popular ontology language, WebODE however, does not have provision for creating OWL ontologies in it, though it supports RDFS and some other ontology languages such as OCML. Whenever a developer is in search of ontology development software or editor, the likes of Protégé might be their best bet.

There are mainly two categories of ontology building tools.

(i) Ontology Development Tools: It means ontology editors that allow users to define new concepts, relations, and instances. These tools usually have capabilities for importing and extending existing ontologies. Development tools generally include graphical browsers, search capabilities, and constraint checking. Protégé 2000, OntoEdit, OilEd, WebODE, and Ontolingua are some examples of development tools.

(ii) Tools for Mapping, Aligning and Merging Ontologies: These are the ontology mapping tools that help users find similarities and differences between source ontologies. Mapping tools either identify potential correspondences automatically or provide the environment for the users to find and define these correspondences, or both. Mapping tools are often extensions of development tools. PROMPT, ONION, Chimaera etc [25].

COMPARATIVE STUDY OF ONTOLOGY DEVELOPMENT TOOLS

The results for comparison of tools are shown in form of Table 1 which are categorized on the basis of following features [26]:

(i) General Issues in which Developers. Release and Pricing Policy are examined.

(ii) Software Architecture in which architecture, extensibility, Ontology Storage and backup facilities are discussed.

(iii) Interoperability in which import and export formats are discussed.

(iv) Knowledge Representation and Methodological support

(v) Inference services in which inference engine and automatic classifications are discussed.

(vi) Usability in which GUI views, working and Ontology library are listed.

CONCLUSION AND FUTURE SCOPE

For Ontology development, effective tools are a central requirement. Fortunately, software tools are already available to achieve most of the required activities of ontology development allowing us to focus specifically on the innovate requirements of ontology development within the model. Projects often involve solutions using numerous ontologies (Wine.rdf, food.owl, Companies.rdf etc.) from external sources. Sometimes there is also the need to use existing and newly developed in-house ontologies(i.e. camera.owl). For this reason it is important that the editing tools for ontology construction promote interoperability.

Ontology Editors prove an asset in the development of ontologies. The need is to identify a suitable editor for a particular domain. A theoretical attempt has been made to analyze and make a comparative analysis of the various ontology editors available and their role in ontology building and maintenance. It can be further extended to choose and make use of an ontology editor for a particular domain ontology creation.

It is quite clear Ontology development is an ad-hoc approach. Among several viable alternatives, one need to find which one would work better for the projected task that can easily and effectively be maintained and expressed. Though foundation of ontology is logic but it is a model of reality and the concepts in the ontology must reflect this reality.. We have described a tool-assisted method for building the basis for ontologies adopted from domain analysis.

Some directions for ongoing and future work include more careful investigation of the reasoning algorithm and its extension to more powerful DL language such as SHIQ (the DL language used by OWL); the study the basic operations needed in reasoning with package and view, such as the construction of default interface and horizon for a package, checking if an entity is in the default interface of a package (visible to the outside); Efficient representation of mapping between packages; Implementation of tools to support P-OWL, such as ontology editor and reasoner

REFERENCES

[1]. Franz Baader, Ian Horrocks, and Ulrike Sattler. Description logics as ontology languages for the semantic web. In Dieter Hutter and Werner Stephan, editors, Festschrift in honor of Jörg Siekmann, Lecture Notes in Artificial Intelligence. Springer, 2003

[2]. Berners-Lee, T., Hendler, J. and Ora Lassila. The semantic web. Scientific American, May 2001.

[3]. Borgida and L. Serafini. Distributed description logics: Directed domain correspondences in federated information sources. In Proceedings of the International Conference on Cooperative Information Systems, 2002.

[4]. Neumann T. and Gerhard W., "RDF3X: a RISC-style Engine for RDF," Proceedings of the VLDB Endowment, vol. 1, no. 1, pp. 647-659, 2008.

[5]. Antoniou G. and Frank V. H., Semantic Web Primer, MIT Press, Cambridge, Massachusetts, London, England, pp. 113-114, 2004.
[6]. Fensel D., Horrocks I., van Harmelen F., Decker S., Erdmann M., and Klein M., Proceedings of the 12th European Workshop on Knowledge Acquisition, Modeling, and Management (EKAW 2000). OIL in a Nutshell. In: Lecture Notes in Artificial Intelligence, pp 1–16, 2000.
[7]. IAEA., 2006. Knowledge Management for Nuclear Industry Operating Organizations, IAEA-TECDOC-1510, 1-185.
[8]. William Hsu, Ricky K. Taira, Suzie El-Saden, Hooshang Kangarloo,and Alex A. T. Bui, "Context-Based Electronic Health Record: Toward Patient Specific Healthcare", IEEE transactions on information technology in biomedicine, 228-234.
[9]. Aditya Kalyanpur, Bijan Parsia, Evren Sirin, Bernardo Cuenca Grau, James A. Hendler: "Swoop: A Web Ontology Editing Browser". J. Web Sem. 4(2): 144-153 (2006).
[10]. C.W. Holsapple and K.D. Joshi, "A collaborative approach to ontology design". Commun. ACM 45 (2002) pp 42–47.
[11]. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press, 2003.
[12]. F. van Harmelen and D. Fensel. Practical Knowledge Representation for the Web. In Proc. IJCAI'99 Workshop on Intelligent Information Integration, 1999
[13]. F. Baader et al. (eds.), The Description Logic Handbook, Cambridge University Press, 2003.
[14]. T. Berners-Lee, J. Hendler, O. Lassila, „The Semantic Web", Scientific American, 5, 2001.
[15]. G. Antoniou and F.V. Harmelen, "Web Ontology Language: OWL", Presented at Handbook on Ontologies, 2004, pp.67-92.
[16]. F. Baader, D. Calvanese, D. McGuinness, D. Nardi and P. Patel-Schneider, "The Description Logic Handbook: Theory, Implementation and Applications", Cambridge University Press, 2003.
[17]. The Syntax of CycL (http://www.cyc.com/cycdoc)
[18]. Sbodio M. L., David M., and Claude M., "Discovering Semantic Web Services Using SPARQL and Intelligent Agents," Elsevier Journal, p. 311, 2010.
[19]. O'Connor M., and Amar D, SQWRL: a Query Language for OWL. In OWL: Experiences and Directions (OWLED), 5th International Workshop pp. 1-8.
[20]. Park J. E., and Sam T. E. H., XML Topic Maps: Creating and Using Topic Maps for Web, Addison Wesley, 2002.
[21]. Mizoguchi, Riichiro, Tutorial on ontological engineering part 2: Ontology development, tools and languages. New Generation Computing Springer, vol.22 (1), pp. 61-96, 2004.
[22]. Noy, Natalya F., and Mark A. Musen, Evaluating ontology-mapping tools: Requirements and experience. Workshop on Evaluation of Ontology Tools at EKAW, vol. 2, 2002.
[23]. Sure, York, et al., OntoEdit: Collaborative ontology development for the semantic web. Springer Berlin Heidelberg, 2002.
[24]. Madurai Meenachi, N. and Sai Baba, M. "A Survey on usage of ontology in various domains", International Journal of Applied Information and Science (IJAIS), 2012, in press.
[25]. Aegis corporation.1995. Introduction to Knowledge Management and Knowledge Base.
[26]. Leonard-Barton, D. and Sensiper. S.1998. "The role of tacit knowledge in group innovation, California management review, 112-125.