Webology, Volume 4, Number 2, June, 2007

Home | Table of Contents | Titles & Subject Index | Authors Index

Overview of Ontology Servers Research

Mohammad Nazir Ahmad
School of Information Technology and Electrical Engineering, the University of Queensland, Australia. E-mail: nazir (at) itee.uq.edu.au

Robert M. Colomb
Faculty of Computer Science and Information Systems, the University of Technology Malaysia, Malaysia. E-mail: colomb (at) utm.my
School of Information Technology and Electrical Engineering, the University of Queensland, Australia. E-mail: colomb (at) itee.uq.edu.au

Received January 30, 2007; Accepted April 25, 2007

Abstract

An ontology is increasingly becoming an essential tool for solving problems in many research areas. The ontology is a complex information object. It can contain millions of concepts in complex relationships. When we want to manage complex information objects, we generally turn to information systems technology. An information system intended to manage ontology is called an ontology server. The ontology server technology is at the time of writing quite immature. Therefore, this paper reviews and compares the main ontology servers that have been reported in the literatures. As a result, we point out several research questions related to server technology.

Keywords

Ontology; Ontology server; Ontology tool; Semantic Web

Introduction

An ontology is an explicit specification of a conceptualization (Gruber, 1993). It is designed artifacts that formally represent agreed semantics of a domain interest in computer resources (Gruber, 1993; Guarino, 1998). This enables the sharing and reuse of information and allows for the interoperation of information systems (Pretorius, 2005). In ontology-based interoperation of information systems, for example, an ontology server is a key information system to support the ontology use. However, research on ontology server at the time of writing is quite immature and most of the ontology servers have taken ad-hoc approach to design and implementation. Despite this research area is well accepted in knowledge representation and Artificial Intelligent community, though there is still no standard approach have been addressed in the design of ontology server, for example. Therefore, to have a better overview of ontology server research may enable us to understand a recent progress in the ontology server technology and then to finding significant gaps for approaching into this research.

Managing Ontologies and Ontology Tool

A major problem with ontologies is how to manage them. Thus, there are related research problems in the field of ontologies which include: (a) creating large-scale ontologies (e.g., Lenat, 1995), (b) defining expressive languages for representing ontological knowledge (e.g., Bechhofer et al., 2004; Lopez et al., 1999) and (c) implementing ontology systems or tools that support ontology-based applications (e.g., Chung et al., 2000; Cui & O'Brien, 2000; Dameron et al., 2004; Farquhar, Fikes & Rice, 1997; Li et al., 2003; Starlab, 2006; Valo, Hyvönen & Komulainen, 2005).

When a new ontology is going to be developed, several basic questions arise related to the methodologies, tools and languages to be used in ontology development as reported in (Corcho, Fernandez-Gomez & Gomez-Perez, 2003). However, the management tasks for ontologies are not only focused on ontology development (design time). Another aspect like how to use ontologies is also important (commit time). We look at these aspects as two fundamental ontology engineering challenges: reusability and usability. The former aspect is main goal in building ontologies. The latter aspect is about how to make use of ontologies. This is an important goal in ontology-based interoperating systems since one of the major uses of ontology is in fact to support interoperation of information systems (Hart et al., 2004) where an exchange can involve thousands of players. We believe that most research focuses on the development of single or integrated tools for supporting ontology development (design time). In other words, they aim to support ontology reusability. The main ontology tools in the mainstream literature for ontology development are well reviewed in (Duineveld et al., 2000; Corcho, Fernandez-Gomez & Gomez-Perez, 2003).

We believe that the notion of ontology server has originally stemmed from the research of ontology development tools. Most works describe implicitly or explicitly the server as a kind of isolated or integrated tool for building ontology (e.g., Eklund, Roberts, & Gree, 2002; Farquhar, Fikes & Rice, 1997; Li et al., 2003; Sure et al., 2002). The isolated tools are not fully integrated with other activities of the ontology life-cycle (Corcho, Fernandez-Gomez & Gomez-Perez, 2003). Such tools are Ontolingua Server (Farquhar, Fikes & Rice, 1997), Ontosaurus (Swartout et al., 1996) and OntoRama (Eklund, Roberts, & Gree, 2002). The latter is much more ambitious, built as robust integrated environment or suite that provides technological support to most of the ontology life-cycle activities. They have extensible, component-based architectures, where modules can easily be added to provide more functionality to the environment (Corcho, Fernandez-Gomez & Gomez-Perez, 2003). Among these environments, we can cite Protégé-2000 (Noy & Musen, 2000), WebODE (Arpirez et al., 2001) and OntoEdit (Sure et al., 2002).

However, the discussion of ontology server in those ontology development tools is quite confusing and still unclear. Some studies discuss the server as ontology repository. In this context, the server is centrally concerned on repository technology (e.g., Harrison & Chan, 2005; Pan, Cranefield & Carter, 2003). Some other studies discuss both ontology repository and server functionality. In this context, the server is described as an information system (e.g., Dameron et al., 2004; Mauger, 2005; Starlab, 2006). Therefore, we would like to highlight three important points about the notion of ontology server discussed in many studies as follows: (a) many studies discuss ontology tools mainly for building ontology. Some of these tools consist of many other modules including the ontology server. In this context, the notion of ontology server is mostly a repository issue. For example, the Ontosaurus consisting of two modules:

ontology server and ontology browser. In other words, the notion of ontology server is referred to ontology repository such as in (Harrison & Chan, 2005). (b) Some studies clearly discuss ontology servers and use the word "ontology server" in their work to refer both server repository and functionalities (e.g., Dameron et al., 2004; Li et al., 2003; Oberle et al., 2002; Starlab, 2006). (c) In some other studies are similar to (b), except the notion of ontology server is referred to ontology repository as mentioned in (a). An example is the Ontolingua server.

We draw three points of view; First, we argue with the notion of ontology server as mentioned in (a) and (c). The notion of ontology server should be referred to its repository and functionalities issues. Second, most of the ontology servers in the literature are developed for building ontologies. In other words, the current studies give emphasis more the ontology server supporting some activities in the ontology building life-cycle. Third, we argue that, to have a clearer understanding in this area, we would need a so-called "framework" to give a brief idea of ontology server research. We point out several basic questions related to the functionalities, repository, and methodological support and so forth to be used in the ontology server development. These questions would include:

> *What are the main components of ontology server? What kind of framework can we use to understand the server? What methodologies can we use for building an ontology server? Does any methodology provide support for building the server? What general activities should we consider when building the server? Which is the life-cycle of an ontology that the server can support? How are the ontologies stored in the server? What are the main server functions? How can applications interoperate with ontology servers and/or use the ontologies? What sort of ontology languages is the server supposed to support? Is the language chosen appropriate for exchanging message between different applications? What is the main purpose of ontology server to be developed? What sort of the ontology server architecture can we propose? And so forth.*

In this paper, we present the main characteristics of ontology servers, which can assist practitioners and researchers in this field to gain answers to the questions above. Since at the time of writing this field is fairly immature, we provide guidelines that outline several opportunities related to ontology server research.

Ontologies and Semantic Web

Ontologies are an important part of the semantic web (Berners-Lee, 1999). The semantic web is an extension of the current web in which information is given well defined meaning, better enabling computers and people to work in cooperation (Berners-Lee, 1999). In the semantic web, ontologies can be used to encode meaning into a web page, so that intelligent agents (applications) can understand what the web page is about, and therefore provide humans with more useful cooperative services (Berners-Lee, 1999). Driven by application needs and the semantic web vision, the ontology server is believed to be a key component for supporting semantic web applications (Agrawal, 2002). Therefore, in this study, we focus only on the ontology servers that support ontology-based applications in open environments such as the Web and semantic web. Furthermore, we look into several ontology servers: (a) are mostly cited in literatures, (b) they explicitly use word "ontology server" in their reports and address server development issues, (c) we assume that several of them are still progressing and (d) focusing on two main aspects of the ontology server: server functionalities and ontology repository.

An Ontology Server

An ontology server is used at design, commit and runtime (Ahmad, Colomb & Cole, 2006). At design time, the ontologist uses the server for developing ontologies. At commit-time, players (i.e. human or applications) commit to a limited part of the ontology to enable the exchange of messages. At runtime, players exchange messages mediated by shared ontologies. Zachman (1987) has proposed an information system architecture framework, which has been well accepted (Ells, 1998; Evernden, 1996; Jones, 2005; O'Rourke, Fishman & Selkow, 2003). There are three key concepts (knowledge, process and communication) and related enabling technologies described by Zachman (1985). Since the ontology server is a kind of information system, it can be described in terms of this framework (see details in (Ahmad, Colomb & Cole, 2006). Based on Zachman's framework, this paper is important to concentrate on the process aspect (server functionality) and knowledge aspects (ontology repository) of the ontology server development.

Ontology Server Functionality

There is a number of different ontology server implementation available. Their functionality focuses on editing, browsing and storing ontologies. In some cases the ontology server also provides an inference engine that allows statements about the relationships between entities in different ontologies to be tested or retrieved (Li et al., 2003). There is a number of facilities the server may provide such as support for creating and editing ontologies, support for publishing and retrieving ontologies (by humans via a graphical user interface and/or via a network protocol), support for recording metadata about ontologies and the relationships between them, support for interactively browsing the structure of ontologies and inference mechanisms to verify the consistency of ontologies.

We would say that there are two main approaches to implementing ontology server functionalities: (a) tool development and (b) Application Programming Interfaces (APIs). The former approach is about developing any kind of application on top of the ontology repository such as ontology browser, ontology editor, ontology translator and so on. In this approach, those tools typically perform a single aspect of ontology server functionality. For example, Starlab (2006) has developed an ontology browser, ontology manager and ontology modeler to managing ontologies stored in the server repository. The API approach can be used to write a program that provides some services to the ontology repository. Typically, any applications or tools can use those APIs to interface the ontology repository. For example, Starlab (2006) has provided some basic database APIs for accessing ontologies stored in the ontology repository. With respect to server functionality, to avoid confusion, we should be aware of many terms appearing in studies of server functions such as server functionality and functions (Starlab, 2006), ontology service (FIPA, 2001), server process (Chung et al., 2000), server services (Farquhar, Fikes & Rice, 1997), server operation and ontology operation (Dameron et al., 2004).

Ontology Server Repository

In this paper, we use term "ontology repository" and "server repository" interchangeably. We would say that the discussion of ontology repository gives emphasis more to repository issues such as storing and organizing ontologies in database. There are such studies discussing specifically ontology repository aspects such as (Harrison & Chan, 2005; Pan, Cranefield & Carter, 2003). For example, in (Harrison & Chan, 2005), they have proposed a generic representation for ontology repository. Similar to (Pan, Cranefield & Carter, 2003), they have introduced a so-called lightweight ontology repository for enabling shareable and maintainable ontologies. A lightweight ontology is referred to a kind of ontology that simple, generic form and does not include axiom that allow deductions to be made. Harrison & Chan (2005) discuss two methods of storing the ontologies. The first method involves storing individual ontology in a separate flat file. The file provides a more straightforward view where its content can be inspected. However, the main problem with this method is that a search engine would need to be developed to search the contents of the ontology files. The second method uses a database to store ontologies. Databases have indexing and other capabilities that enable faster searching. Therefore, to ensure scalability and maintainable large ontologies, the database method rather than a flat file we think would preferable for storing ontologies. However, how the ontologies are stored depends on how the ontology representation should appear in the server and what types of database are considered (i.e. relational, object-oriented or object-relational).

A Comparison of Ontology Servers

After an extensive search on the Internet and several journals and conferences, and with several focus as stated in the section entitled "*Ontologies and Semantic Web*", we selected the following eight ontology servers: Ontolingua server (Farquhar, Fikes & Rice, 1997), ACOS (Li et al., 2003), Starlab (2006), KAON (Oberle et al., 2002), OntoRama (Eklund, Roberts, & Gree, 2002), OWS (Dameron et al., 2004), FIPA server (2001), and Adapted Ontology Server (Chung et al., 2000).

The Ontolingua server has been running since 1995 (Farquhar, Fikes & Rice, 1997). It has been developed in the Knowledge Systems Laboratory (KSL) at Stanford University. It is a tool that supports distributed, collaborative editing, browsing and creation of Ontolingua ontologies with a form-based web application. Remote editors can browse and edit ontologies, and remote or local applications can access any of the ontologies in the ontology repository with the OKBC protocol. The Ontolingua ontology uses the representation languages, Ontolingua Frame Ontology and KIF, which are wide spectrum language capable of representing fine features of concepts (i.e. are based on description logics). The Ontolingua server supports ontology inclusion and circular dependencies. Its consistency-check capability, however, is restricted to the functions similar to database schema checking (Li et al., 2003). For instance, "all slots, slot values, facets and facets values are checked to make sure that they conform to the constraints that they apply (i.e. domain, range, slot value type, and cardinality constraints)". This means that semantic consistency checking is done in domain experts' heads. It is a large project focusing on ontology development. It has built sophisticated tools for developing and maintaining frame-based ontologies. However, it does not address problems related to legacy systems and tools to merge ontologies (Cui & O'Brien, 2000). The users interact with the single server through a web browser to create, edit and browse ontologies. Users have to tolerate the network delays and server response delays. The Ontolingua server uses Ontolingua-based repository for storing ontologies and developing a set of tools for demonstrating server functions.

The second server is ACOS. It has been developed in the Intelligent System Laboratory (ISLab) at British Telecom Research (Li et al., 2003). Li et al. (2003) claim it to be a community-oriented ontology server. It provides the way for the so-called "community" to construct ontologies. The concept of "community" in ontology management enables everyone to have the opportunities of influencing ontology construction. This depends on the users' importance score, and such score is computed mechanically based on how active these users are in contributing to the knowledge base. They came up this kind of vision because in an open environment, ontology is the asset of all participants; every user can join, contribute and leave such community. The opposite of this approach is called "central". The central-controlled mechanism would not appear to be appropriate for this situation (Li et al., 2003). However, most of the ontology server implementations in many studies apply the "central" approach to ontology development (e.g., Chung et al., 2000; Eklund, Roberts, & Gree, 2002; Farquhar, Fikes & Rice, 1997; Starlab, 2006). The "central" means a centralized ontology server, only a fixed group of users have the rights to modify ontologies, which is similar to the situation in database management (Li et al., 2003). Similar to Ontolingua server, the ACOS has also provided consistency checking but not limited to humans. A wide range of software agent communities also can share it and the server facilitates on-line ontology construction, consistency-check and use. Li et al. (2003) claim that ACOS is designed to be an online community in which a diverse group of software agents can contribute and use ontologies at runtime. A key enabler in this scenario is a high degree of "shareness" of the ontologies maintained by the server. This relates to designing an appropriate knowledge representation, which is the first step towards building an ontology server. In this context, Li et al. (2003) believe that to achieve "shareness", it is required that local features of ontology are removed. Examples of constructors that can bring in local features include *property* in DAML-OIL or *slot* in Ontolingua. Other examples of constructors that bring in local effect are *part-of* in Framework and *disjoint-with* in Descriptions Logic. In other words, Li et al. (2003) accept the shared ontologies should have minimal expressiveness, which consist of a minimal set of axioms written in a language of minimal expressivities. Thus, to achieve a high degree of shareness, Li et al. (2003) believe that constructors *Class, subClassOf, SameClassAs, SuperClassOf* and *InstanceOf* offer more stability in the process evolution and minimal expressivities. Compared to Ontolingua server, the ACOS uses DAML-OIL as its ontology representation language. The ACOS server uses file-based repository for storing DAML-OIL ontologies and developing a set of APIs for implementing server functions. To facilitate collaborative ontology development, this server implemented an import mechanism that is similar to the *inclusion model* in Ontolingua server and *extension relationship* in FIPA ontology server (FIPA, 2001).

Third is the Starlab. This server is under development and still in progress several deliverables. This research initially falls within the DOGMA research framework (Jarrar & Meersman, 2002). At the time of writing, we would say that this is a new ontology server research project, is funded for 5 years by the Vrije Universitet Brussel. The mission of this server is to assist the gathering and incremental growth of ontologies (Starlab, 2006). In terms of ontology representation, this server is in line with the rationale of ACOS server previously discussed. The proposed ontology model consists of five basic elements: context, terms, concepts, roles and lexicon. Starlab (2006) claims that constraints and derivation rules are intentionally left outside the ontology. At the moment, they are still experimenting with the ontology model and an early version of ontology server has been implemented. In the first prototype, consistency-check and user control are not included (Starlab, 2006). This server uses MSQL server to store the ontology and ontology objects are expressed in XML. Its server functions are implemented based on developed tools and basic database APIs. In addition, the server's ontology representation also benefits from graphical notation based on Object Role Modeling (ORM) languages.

Fourth is the KAON server developed by AIFB in Karlsruhe University. Similarly to Ontolingua server and Starlab, the KAON server is a result of sophisticated tools delivered by a large research project conducted at the University of Karlsruhe. This server provides its functions through a set of APIs and store RDF-based ontologies using a relational database (Maedche et al., 2003; Oberle et al., 2002). It is an early prototype of an ontology-based application server. In essence, it uses JBoss and extends by the tools of the KAON tool suite for reasoning with software components (Oberle et al., 2002). The goal is to support the developer in his daily tasks with reasoning. In terms of ontology representation, it is similar to Ontolingua server: the complexity of ontology is not left outside the ontology, which contrasts with Starlab and ACOS.

Fifth is the OntoRama server was developed in (Eklund, Roberts & Gree, 2002). The main functions of OntoRama include search, compare and modify WebKB ontologies. It does not support consistency-check or cross-ontology queries; as a result, its capability of supporting online collaborative ontology construction is restricted (Li et al., 2003). In our opinion, this server demonstrates functions of storing and browsing RDF-based ontologies, while supporting ontology development is not its main purpose.

Sixth, the OWS server proposed by Dameron et al. (2004) at the Stanford Medical Informatics (SMI), Stanford University. This server does not generally deal with ontology development. It is used for all other tasks, assuming ontologies already exist. The implementation of this server is bit tricky. In fact, the server described by Dameron et al. (2004) does not itself serve as ontology repository (i.e. the server itself does not store the ontologies). The server provides services, taking ontologies as inputs. Hence, they claim that ontology evolution is not a problem here, because a change simply gives the server a new ontology as inputs. In our opinion, we would say that this server supports runtime interoperation. Server functions are implemented using web services assuming that ontologies naturally reside in any player's sites. The ontology storage is file-based, which are held on different players' sites.

Seventh is the FIPA ontology server. The FIPA (Foundation for Physical Agents) is a standards body, which has developed interaction standards for agents in open environment. In the FIPA's ontology server specification, its server's ontological representation is divided into a *fine-grained ontology*, called heavyweight ontology in (Pretorius, 2005) and a coarse ontology that consists of a minimal set of axioms written in a language of minimal expressivity, called lightweight ontology in (Pretorius, 2005). However, the FIPA's server supports these two kinds of ontology representation but with different scope of use and level of detail (FIPA, 2001).

Eighth is the server proposed by Chung et al. (2000) at SungKyunKwan University, Korea. This server is intended specifically for ontology developed for electronic commerce applications. In terms of ontology representation, Chung et al. (2000) define two main criteria for ontology: (a) ontology can be translated and (b) ontology should be practical. Its server functions include gathering information from the Web, creating a relation, modifying and rebuilding the standard ontology and servicing the standard ontology. This server uses a MySQL database to store ontologies and Java APIs for implementing its server functions. We assume that this server allows medium ontological representation. This because the standard ontology is built from the local terms used in sites, and then the server provides an editor tool for making relation between ontologies. In addition, they claim that a standard ontology necessarily has the objective and concrete property. In general, this server supports ontology development but is limited to electronic commerce and deals with the current web.

Comparison Framework

A survey of ontology servers was firstly done in (Li et al., 2003). However, Li et al (2003) did not include several key ontology servers such as KAON and OWS in their survey. In addition, the proposed comparison is too general and does not highlight other important aspects of development issues such as a standard development approach, methodological support and the use of graphical modeling language. In this study, we have compared ontology servers with respect to the similar criteria to (Li et al, 2003), extending some of them, and also adopting several dimensions used in (Corcho, Fernandez-Gomez & Gomez-Perez, 2003; Duineveld et al., 2000). Consequently, we incorporate all these criteria into five dimensions as follows:

> ***General****: It refers to the generic aspects of the server development such as: What is the main goal of developed ontology server? Who are those developers? Can we access to those servers? What is the main phase that server can support? What are the main deliverables of the server development? Which kinds of management provision does the server have? Does the server support collaborative environment? What is the status of the server?* ***Ontology****: This dimension refers to the general questions about ontology representation that the server supports, such as: What the main ontology engineering challenges that the server mostly focuses on? What types of ontology can the server support? What does knowledge representation in server repository look like? What ontology language is the server based on? How expressive is language? Does the server use all the language features (i.e. ontological constructors) to represent shared ontologies?* ***Features****: It represents general features of ontology server in terms of: What are the main server functions have been developed? What types of repository platform the server use to store ontologies? What is the architecture of the server? Does the server provide extensibility?* ***Implementation****: This dimension addresses some aspects of the server technology including: How is the server functionality implemented? What type of technology platform does the server use and support? Does the server use a standard technology? What type of technology is used to access to the ontology repository?* ***Methodology****: This dimension concerning the methodology related to the ontology server development such as: Is there any methods or techniques support for building the server? Is there any ontology-related methodology support for the ontology? To have a clearer understanding of this study, as a result, we summarize this comparative study in Table 1 (see appendix).*

Conclusions and Future Works

We point out some important points based on a Table 1. First, we see that most ontology servers available mostly focus on design time (see *main phase of ontology life-cycle*). It gives supports more to ontology development and tends assume its users to be ontologists. Its main goal is reusability (see *main ontology engineering challenges*) since it saves time and cost. Furthermore, it is well accepted to investigate the wider notion of ontology modularization as mechanism to enable an ontology reuse for developing ontology (see *Ontology-related method or techniques*). There are several approaches in ontology modularization in which we believe they are determined by what ontology is made for and are based on different principle and rely on different assumptions. For example, the ontologies supporting runtime interoperation, which used in business application, are slightly different from ontologies for supporting engineering applications (Colomb et al., 2006; Hart et al., 2004). Therefore, we should first understand the intended use of the ontology which then leads to how it is engineered and then how the supporting server is going to be developed. Second, surprisingly, none of the existing ontology servers are discussed in the context of commit-time issue (see *main phase of ontology life-cycle*). Although in general we have a design and runtime, but the issue how players can be helped to see the parts of ontology before joining the exchange should be firstly addressed. Third, a standard methodological approach for designing ontology and ontology servers is still missing (see *a methodology dimension*). Therefore, there is no server modeling profiles that can be referenced as useful example to guide a developer on how to develop ontology server. However, recently, there have been initiatives to bridge MDA (Model Driven Architecture) and ontologies. Many researchers suggest the use of Unified Modeling Languages (UML) in ontology development (e.g., Baclawski et al., 2002; Colomb et al., 2006; Cranefield & Purvis, 1999; Gasevic, Djuric & Devedzic, 2005; Kogut et al., 2002;). Thus, it would be advantageous to consider these works in the context of ontology and ontology server development. For example, it would be useful to benefit from the standard graphical ontological representations like UML. Unfortunately, most of the ontology and server developed do not consider the use of a standard graphical language.

Fourth, there is almost no explicit discussion of servers for supporting information systems interoperation in the mainstream literatures. So, the use of server as a runtime tool is not well developed. Thus, we would say, the server functions to support many aspects of the semantic web are still missing. Fifth, there are issues in ontological representation that relate to ontological repository. Most of the servers do not use database to store ontologies. However, we believe that most of the servers are moving towards using database and Java platforms (see *implementation*). Sixth, generally, there are two types of ontologies; lightweight and heavyweight (see *types of ontologies*). The former simply sees ontology as a description with the aim of organizing concepts. The latter defines ontology as a complete theory consisting of both a formal vocabulary and defined axioms that allow further deductions or inferences to be made. Most of the servers are influenced by lightweight ontologies. These servers only support using minimal language constructors, which have very minimal expressivity of ontology precision (i.e. *class, subClassof, IsA*). The delivered ontology from a knowledge representation point of view looks simple compare to heavyweight ontology. However, we argue that this situation is also determined by what the ontology is made for. For example, ontology supporting information systems interoperation needs heavyweight ontology that richer and can be inferred by software agents.

From the perspective of a survey study, this review reports the current technologies of ontology and the semantic web, particularly on ontology server development. Although we do not provide complete technical details, it is novel enough to illustrate opportunities for approaching automated ontology systems primarily the ontology server development. We know that ontology server is built around database core but considering both server functionalities and server repository lead to treat the server is a kind of information systems that can benefit from other field like software engineering. Our future work is to investigate a useful methodological approach for designing ontology and ontology server in the context of commit-time issues. In this case, there is a commit-time scenario which has several commit-time requirements need to be addressed as discussed in (Ahmad & Colomb, 2007). Briefly, in a concrete point of view, we have an ontology and there is the concept of *players* which refers to a *user* responsible for an application that is to commit to the ontology. The ontology consists of a collection of objects (i.e. classes, individuals, properties, property instances). The user employs the *browser* to identify the objects the user considers needed to commit the application. We argue that this should importantly be addressed since in the complex and large-scale ontologies, most of the players (i.e. human or software agents) are directly interested in the portion of the ontologies before joining some exchanges.

References

- Ahmad, M. N., & Colomb, R.M. (2007). Commit Time Requirements for an Ontology Server. Paper presented at the *Pacific-Asia Conference on Information Systems* (PACIS'07), Auckland, New Zealand, 03-07 July.
- Ahmad, M. N., Colomb, R.M., & Cole, J. (2006). An Ontology Server: A Knowledge Tool for Systems Interoperability in Semantic Web Context. *Journal of Information Technology*, 18(1), 1-23.
- Agrawal, R. (2002). Making Semantic Web Real: Some Building Blocks. Paper presented at the *International Workshop on Semantic Web*, Hawaii.
- Arpirez, J. C., Corcho, O., Fernandez-Lopez, M., & Gomez-Perez, A. (2001). WebODE: A scalable ontological engineering workbench. Paper presented at the *First International Conference on Knowledge Capture* (KCAP_01), Victoria.
- Baclawski, K., Kokar, M., Kogut, P., Hart, L., Smith, J., Holmes, W., Letkowski, J., Aronson, M., & Emery, P. (2002). Extending the UML for Ontology Development. *International Journal of Sofware and Systems Modeling*, 1(2), 142-156.
- Bechhofer, S., Van Harmelen, F., Hendler, J., Horrocks, I., McGuiness, D.L. Patel-Schneider, P.F. & Stein, L.A. (2004). OWL Web Ontology Language Reference. W3C Recommendation. Retrieved July 26, 2006 from http://www.w3.org/TR/2004/REC-owl-ref-20040210/
- Berners-Lee, T. (1999). *Weaving the Web*. London: Orion Business Books.
- Chung, H., Choi, J., Yi, J., Han, J., & Lee, E. (2000). A Construction of the Adapted Ontology Server in EC. *Lecture Notes in Computer Science*, 1983, 355-360.
- Colomb, R. M., Raymond, K., Hart, L., Emery, P., Welty, C., Xie, G.T., & Kendall, E. (2006). Version 3.3: The Object Management Group Ontology Definition Metamodel. In Coral, C., Francisco, R., & Mario P. (Ed.), *Ontologies for Software Engineering and Software Technology* (pp. 1-25): Springer.
- Corcho, O., Fernandez-Lopez, M., Gomez-Perez, A. (2003). Methodologies, Tools and Languages for Building Ontologies. Where is their meeting point? Journal of Data and Knowledge Engineering. *Journal of Data and Knowledge Engineering*, 46(41).
- Cranefield, S., & Purvis, M. (1999). UML as an Ontology Modeling Language. Paper presented at the *Workshop on Intelligent Information Integration, 16th International Joint Conference on Artificial Intelligence*, IJCAI-99.
- Cui, Z., & O'Brien, P. (2000). Domain Ontology Management Environment. *Proceedings of the 33rd Hawaii International Conference on System Sciences*, IEEE.
- Dameron, O., Noy, N.F., Knublauch, H. & Musen, M.A. (2004). Accessing and Manipulating Ontologies Using Web Services. Paper presented at the *Workshop on Semantic Web Services: Preparing to Meet the World of Business Applications* at the *Third International Conference on the Semantic Web* (ISWC-2004), Hiroshima, Japan.
- Duineveld, A. J., Stoter, R., Weiden, M.R., Kenepa, B. & Benjamins, V.R. (2000). WonderTools? A comparative study of ontological engineering tools. *International Journal of Human-Computer Studies*, 52, 1111-1133.
- Eklund, P., Roberts, N., & Gree, S. (2002). OntoRama: Browsing RDF Ontologies using a Hyperbolic-style Browser. Paper presented at the *First International Symposium on Cyber Worlds* (CW.02), IEEE Computer Society.
- Ells, R. (1988). Hypermedia, Help and How-To. *Proceedings of the 16th annual ACM SIGUCCS Conference on User Services*, Long Beach, California, United States.
- Evernden, R. (1996). The Information Framework. *IBM Systems Journal*, 35(1), 37-68.
- Farquhar, A., Fikes, R. & Rice, J. (1997). The Ontolingua Server: A Tool for Collaborative Ontology Constructions. *International Journal of Human-Computer Studies*, 46(6), 707-728.
- FIPA. (2001). Foundation for Intelligent Physical Agents, Specification of 2001, FIPA Ontology Service Specification, Retrieved 23 July 2006 from http://www.fipa.org/specs/fipa00086/index.html
- Gruber, T. R. (1993). A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition*, 5(2), 199-220.
- Gasevic, D. V., Djuric, D.O. and Devedzic, V.B. (2005). Bridging MDA and OWL Ontologies. *Journal of Web Engineering*, 4(2), 118-143.
- Guarino, N. (1998). Formal Ontology in Informations Systems. *Proceeding of the 1st International Conference on Formal Ontology in Information Systems* (FOIS'98), Trento, Italy.
- Hart, L., Emery, P., Colomb, R., Raymond, K., Chang, D., Ye, Y., Kendall, E. & Dutra, M. (2004). Usage Scenarios and Goals for Ontology Definition Metamodel. *Lecture Notes in Computer Science*, 3306, 596.
- Harrison, R., & Chan, C.W. (2005). Distributed Ontology Management System. *Proceedings of the 18th Annual Canadian Conference on Electrical and Computer Engineering*, (CCECE'05), IEEE, Saskatoon.
- Jarrar, M., & Meersman, R. (2002). Formal Ontology Engineering in the DOGMA Approach. *Proceeding of ODBASE'02*, LNCS.
- Jones, S. (2005). Toward an Acceptable Definition of Service. *IEEE Software*, 22(3), 87-93.
- Kogut, P., Cranfield, S., Hart, L., Dutra, M., Baclawski, K., Kokar, M., & Smith, J. (2002). UML for Ontology Development. *The Knowledge Engineering Review*, 17(1), 61-64.
- Lenat, D. B. (1995). CYC: A Large-scale investment in knowledge infrastructure. *Communication of ACM*, 38(11), 33-38.
- Li, Y., Thompson, S., Tan, Z., Giles, N., & Gharib, H. (2003). Beyond Ontology Construction: Ontology Services as Online Knowledge Sharing Communities. *Lecture Notes in Computer Science*, Volume 2870, 469-483.
- Lopez, M. F., Gomez-Perez, A., Sierra, J.P., & Sierra, A.P. (1999). Building a Chemical Ontology using Methontology and the Ontology Design Environment. *IEEE Intelligent Systems*, 14(1), 37-46.
- Mauger, B. (2005). *Tools to Help an Agent Commit to an Ontology*. Undergraduate Thesis, University of Queensland.
- Maedche, A., Motik, B., Stojanovic, L. (2003). Managing Multiple and Distributed Ontologies on the Semantic Web. *The VLDB Journal*, 12(4), 286-302.
- Noy, N. F., & Musen, M. (2000). PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. *Proceedings of the Seventeenth National Conference on Artificial Intelligence* (AAAI-2000), Austin, TX.
- O'Rourke, C., Fishman, N., & Selkow, W. (2003). *Enterprise Architecture Using the Zachman Framework*. USA: Course Technology.
- Oberle, D., Volz, R., Motik, B., & Staab, S. (2002). *KAON Server Prototype. Technical Report, IST Project 2001-33052 WonderWeb*, Department of Computer Science, The Victoria University of Manchester.
- Pan, J., Cranefield, S. & Carter, D. (2003). A Lightweight Ontology Repository. Paper presented at the *AAMAS'03*, ACM, Melbourne, Australia.
- Pretorius, A. J. (2005). Visual Analysis for Ontology Engineering. *Journal of Visual Languages and Computing*, 16, 359-381.
- Sure, Y., Erdmann, M., Angele, J., Staab, S., Studer, R. & Wenke, D. (2002). OntoEdit: Collaborative Ontology Engineering for the Semantic Web. *Proceedings of the 1st International Semantic Web Conference*, LNCS 2342, Sardinia, Italy.
- Swartout, B., Patil, R., Knight, K. & Russ, T. (1996). Towards Distributed use of large-scale ontologies. *Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, Banff, Canada.
- Star Lab. V.U.B. Ontology Server Research. Retrieved 3 May 2006 from http://www.starlab.vub.ac.be/research/dogma/OntologyServer.htm
- Valo, A., Hyvönen, A. & Komulainen, V. (2005.). Tool for Collaborative Ontology Development for the Semantic Web. *Proceedings of International Conference on Dublin Core and Metadata Applications* (DC 2005).
- Zachman, J. A. (1987). A Framework for Information Systems Architecture. *IBM Systems Journal*, 26(3), 276-292.

(Appendix) Table 1. A Comparison of Ontology Servers

	Ontolingua	ACOS	Starlab	KAON	OntoRama	OWS	FIPA Ontology Service Specification	Adap Onto Serv
General								
Main goal	Collaborative ontology development	Collaborative ontology development	Ontology development (i.e. gathering and incremental growth)	Ontology development	Storing, browsing ontology. Limited support to ontology development	Accessing and manipulating ontologies	Ontology development and ontology-based agent Communication	Onto deve speci in ele comr
Developers	KSL	ISLab	STARLab	AIFB	ITEE and DSTC	SMI	FIPA	SECE
Release	Free web access	Not mentioned	Free web access	Open source	Free web access	No	Updated in 2001	No
Main phase of ontology life-cycle	Design time	Design time	Design time	Design time	Design time	Runtime	Design time, Runtime	Desi
Main deliverables	Server functions mainly for supporting ontology development	Server functions mainly for supporting ontology development	Server functions mainly for supporting ontology development	Server functions mainly for supporting ontology development	Server functions mainly for browsing ontology	Server functions mainly for accessing and manipulating ontologies	Server functions mainly for ontology development and agent communication	Serv funct main supp ontol deve
Management provision	Central	Community	Central	Central	Central	Central - (Tasks)	N/A	Cent
Collaborative (i.e. building, access and manipulating, use)	Yes	Yes	Not Mentioned	Yes	No	Yes	Not mentioned	Not ment
Overall remarks	Lack of extensibility, strictly oriented to research activities.	Lack of extensibility, strictly oriented to research activities.	Lack of extensibility, strictly oriented to research activities.	To support integrated environment and much more ambitious.	Lack of extensibility, strictly oriented to research activities.	Lack of extensibility, strictly oriented to research activities.	Provide standards ontology service that used as guidance in specifying ontology server functions.	Strict orien resea activ and c not s towa sema web
Status	Rolled out in 1995	Rolled out in 2002	Being Developed	Rolled out in 2001	Rolled out in 2002	Being Developed	Specification 2001	Rolle 2000
Ontology								
Main ontology engineering challenges	Reusability	Reusability	Reusability	Reusability	Viewing	Usability	Reusability	Extra Reus
Type of ontologies	Heavyweight	Lightweight	Lightweight	Heavyweight	Lightweight	Heavyweight	Heavyweight, Lightweight	Light
Knowledge representation	Complex	Simple	Medium	Complex	Medium	Open (unspecified)	Complex	Medi
Ontology model (i.e. expressiveness	Allow maximal expressiveness	Allow minimal expressiveness	Allow minimal expressiveness	Allow maximal expressiveness	Not mentioned	Not mentioned	Open (unspecified)	Not ment
Ontology languages	Ontolingua and KIF	DAML+OIL	XML	RDF	RDF	Open (unspecified)	Open (unspecified)	Not ment
Features								
Import mechanism	Yes	Yes	No	Yes	No	No	Yes	No
Manner of consistency checking	Human Approval, Simple Data Type Checking	Auto	Not mentioned	Yes	Not mentioned	Semi-auto	Open (unspecified)	Hum
User access control	Yes	Yes	No	Yes	No	Yes	N/A	Not ment
Ontology repository platform	File-Based	File-based	DBMS	File-based + DBMS	File-Based	Open (unspecified)	Open (unspecified)	DBM
Software architecture (i.e. client-server, n-tier application, etc)	Client-server	Client-server	Client-server	Client-server	Client-server	Client-server	Client-server	Clien
Extensibility (i.e. plug- in features, etc)	No	No	No	Yes	No	No	Not Mentioned	No
Implementation								
Server functionality implementation	Tool development (e.g., editor)	APIs (e.g., database access)	Database APIs + Tool development	APIs + Tool development (e.g.,	Tool development (e.g., browser)	Semantic web services	Tool development + APIs	APIs

			(e.g., database access, browser, manager)	database access, editor)				
Server functionality platform	Not Mentioned	Java	Java	Java	Java	Java	Open (unspecified)	Java
User / Application remote support to ontologies (i.e. protocol used)	HTTP /OKBC	HTTP/ JDBC-ODBC	JDBC-ODBC	HTTP/JDBC-ODBC	HTTP/ JDBC-ODBC	HTTP/ JDBC-ODBC	HTTP/ OKBC	HTTP ODB
Ontology repository technology	Ontolingua-based repository	Jena + DAML + OIL	MSQL Server	RDF + RDBMS	Jena + RDF	Open (unspecified)	Open (unspecified)	MyS
Methodology								
Ontology-related method or techniques	Ontology modularization (e.g. structural relationships	Ontology modularization(e.g. ontology connection graph)	Ontology double articulation and ontology modularization	Not mentioned	Ontology modularization(e.g. structural relationships)	Not mentioned	METHONTOLOGY (e.g. guided by ontology development methodology)	Not ment
Server development methodology	No - Ad Hoc	No - Ad Hoc	No - Ad Hoc	No - Ad Hoc	No - Ad Hoc	No - Ad Hoc	No - Ad Hoc	No -
Benefit from graphical language	No	No	Yes - ORM for ontology modeling	No	Yes - Hyperbolic style for browsing ontologies	No	No	No
Server design profiles	No	No	No	No	No	No	No	No

Bibliographic information of this paper for citing:

Ahmad, Mohammad Nazir, & Colomb, Robert M. (2007). "Overview of Ontology Servers Research." *Webology*, **4**(2), Article 43. Available at: http://www.webology.org/2007/v4n2/a43.html

Alert us when: New articles cite this article