Ontology Library
A New Approach for Storing, Searching and Discovering Ontologies

Daniel Kotowski and Deborah A. Stacey
School of Computer Science, University of Guelph, 50 Stone Rd. E., Guelph, Canada

Keywords: Ontology, Ontology Reuse, Ontology Repository, Ontology Library, Knowledge Sharing, Knowledge Engineering.

Abstract: The backbone of semantic web technologies is the ontology. This is a powerful structure, which allows for the capture, reasoning and storing of expert knowledge across various domains. Ideally these structures should be developed and implemented by experts in a set domain as well as designed with re-usability in mind. However, often due to the lack of availability and difficulties of discovering ontologies, these structures are repeatedly recreated. Current methods for storing, discovering and sharing ontologies employ similar techniques as to those used for software source code or static web pages. These are exposed to the limitation inherent with keyword-based searches, such as ambiguity with the keywords themselves and therefore, the most relevant ontology may not be discovered. This paper will examine some of the existing techniques used for the storing and sharing of ontologies. It will offer a contrasting method analogous to software libraries to develop a standard to store, share, discover, and distribute common ontologies.

1 INTRODUCTION

The applicability of ontologies stretches across several domains. Ontologies not only describe but also reason across data which is a powerful tool for developers and data mining. However, these structures perform a complex task and are inherently complex to produce, validate, discover, and distribute. Although it may not be possible to decrease the difficulty of creation and validation of these knowledge artefacts, it is possible to decrease the complexity of the discovery and distribution of ontologies.

Although the semantic web initiative has helped bolster the use of this technology, it has yet to provide a solution for easy search and distribution of ontologies. This is crucial for ontologies to reach critical mass. Considering the cost of entry and difficulty of designing and validating an ontology most users may not choose to use them. To reduce the cost of entry substantially, ontologies need a tool to make them complete, validated, reputable and of high availability.

In this paper we will examine, several existing ontology repository projects with emphasis on discovery, reuse and distribution of ontologies for an average user. We will also define for whom we are designing this library (user-centred design). When examining the existing ontology repositories we will examine them against the defined general user and the key factors they are looking for.

This paper will look at repositories which specifically focus on the discovery and distribution of ontologies. What are the key gaps that exist. As well we will purpose a method analogous to software libraries and describe why this metaphor works so well with ontologies.

2 ONTOLOGIES, THEIR DESIGN AND THEIR USERS

In this section we will examine what ontologies are and how they are developed. This will give us context for these structures and their unique properties along with insight as to how to actually distribute them.

2.1 Ontologies and Ontology Engineering

Ontologies are defined as "a formal, explicit specification of a shared conceptualization. Conceptualization refers to an abstract model of phenomena in the world by having identified the relevant concepts

Kotowski D. and A. Stacey D..
Ontology Library - A New Approach for Storing, Searching and Discovering Ontologies.
DOI: 10.5220/0004145702710277
In *Proceedings of the International Conference on Knowledge Engineering and Ontology Development* (KEOD-2012), pages 271-277
ISBN: 978-989-8565-30-3

of those phenomena. Explicit means that the type of concepts used, and the constraints on their use are explicitly defined. Formal refers to the fact that the ontology should be machine readable. Shared reflects that ontology should capture consensual knowledge accepted by the communities " (Gruber, 1995) (Cardoso and Sheth, 2005).

The usefulness of the ontology can be clearly seen from its various characteristics. However one aspect which has great importance is that the conceptualization must be shared. Thus when designing the ontology concepts that are being described and the relations established between them, the ontology concepts must be agreed upon and understood by multiple stake holders, or the ontology itself will have little to no use (Guarino et al., 2009).

Much of the ontology life cycle is like that of the software engineering life cycle. It follows the path of assessment, implementation, review, refinement (Sure et al., 2009). During the review and refinement stages, the ontologies are often accessed both for the knowledge they are representing and the relations they express, much like metrics applied to software during the verification and validation stages. The parallels between ontologies and software is extremely apparent in this engineering model.

2.2 Designed for Reuse

As described in one of the seminal papers on ontoogies (Gruber, 1995), often when designing ontologies they are designed with reuse in mind. Much of the engineering life cycle presented in (Sure et al., 2009), focusses on the ability to use the ontologies for multiple purposes and the notion of generalization when applying an ontology to various domains. Considering the time and complexity put into developing these structures, it is wise to reuse them as much as possible. "Furthermore, in cases where an ontology is reused, (e.g. as the basis for building a new ontology rather than starting from scratch) descriptions of how the ontologies are applied are terse or absent" (Uschold et al., 1998). This represents an obvious gap in the technology.

There has also been work done developing frameworks for reuse such as that seen in (Gillespie et al., 2011). Here the authors developed a framework to design ontologies that can be reused across different types of ontology driven compositional systems.

2.3 Who is the User?

It is important to understand that an ontology user may not be an ontology expert. It is not necessary for a user of software to be an expert programer, and the same holds true for an ontology. Ontologies are a complex structure to design and implement; once established any user should be able to query them for information. It should not take an expert to understand the intricate nature of an ontology to use one. Many software developers understand how to call a system function but may not necessarily understand how the call functions; the same is true for ontologies, as a user should be able to query an ontology without knowing every class which exists in one.

Some users are interested in extending existing ontologies to meet their needs, and will require more in-depth understanding of how the ontology is constructed. This could be provided through extensive documentation for an ontology. A user may also want to know the intellectual rights of an/group of ontologies to assess weither or not they can freely share them within their project. This information is as important as the ontologies themselves.

3 ONTOLOGY DISTRIBUTION

As described in the sections previous, it is difficult to design, engineer, and validate an ontology. Ideally a user should be able to search for and reuse existing ontologies. However currently it is not clear how to locate ontologies and even if an appropriate ontology was found, how should the user reuse it (Maedche et al., 2003). "A breakthrough in ontology technology would require methodological aids and tools that enable effective and efficient development. A key aspect in achieving this is successful re-use of ontologies" (Ding and Fensel, 2001). This can be achieved through either a library or a repository.

"An Ontology library system is a library system that offers various functions for managing, adapting and standardizing groups of ontologies. It should fulfill the needs for re-use of ontologies. In this sense, an ontology library system should be easily accessible and offer efficient support for re-using existing relevant ontologies and standardizing them based on upper-level ontologies and ontology representation languages" (Ding and Fensel, 2001). Though this definition encapsulates a majority of what is required in ontology library systems there is other information needed to satisfy all requirements as described above.

Additional aspects which should also be considered when developing such a system, are the notions of trust, quality, and information on who published the ontology. When using ontologies for important life critical tasks such health and medicine it is important to understand that the creator of the ontology was qualified to capture said knowledge. For instance, you would not want a history professional designing an ontology on the intricate interactions between cancer cells; it would be much more preferable that the ontology was published by an organization or individual who is actively researching cancer.

Versioning information should include information such as what the nature of the change was from the previous version to understand if the change was an addition, a subtraction, or worse yet a modification of existing relations. This is incredibly important when considering whether or not to update the current ontology being used to a newer version. If the ontology is critical to the system's functionality it may be wise to hesitate before adopting the newest version of the ontology.

The last and most important aspect to consider when designing any type of ontology distribution system is the user. As described above, the user should be made central in any of these system implementation, as the key goal for any library is for mass adoption and by targeting the user we will be able to have wider adoption of these types of systems.

3.1 Current Methods of Distribution

In this section we will examine existing ontology repositories. We will take note of the features they offer and critique them against the needs described in the section above.

3.1.1 BioPortal

BioPortal is a project that aims to store and distribute biomedical ontologies (Noy et al., 2009). These ontologies are produced by the biomedical community and uploaded to this application through a web portal. One of the key goals of BioPortal is to create an open access repository where users can upload new ontologies, edit and annotate current ontologies as well as provide mappings between existing ontologies within the repository. Additionally, users can push changes and notes to current users of the ontology.

BioPortal implements a search feature where, "Users can search for terms within an ontology or across all BioPortal ontologies. Searches can be restricted to class names, properties or other attributes. Searches can also be based on exact matches or soundex. In addition, users can search ontology metadata to find particular types of ontologies. BioPortal contains a master index of all ontology content and metadata to streamline these searches" (Rubin et al., 2007).

Since BioPortal heavily relies on the community for maintenance, support, mapping, and upkeep of the ontologoies, the only notion of quality is via a crowd sourced rating system. Users who have used an ontology found on BioPortal can upload the project it was used on and write comments/review (Noy et al., 2009). This puts an incredible onus on the projects which use an ontology to take time out of their development cycle and leave comments about the ontology. As with many of the ontologies found on BioPortal, currently most do not have any ratings or reviews associated with them.

3.1.2 SOR

SOR (Scalable Ontology Repository) is a system designed to store, search, and reason against many ontologies (Lu et al., 2007). It uses a relational database to store the ontologies and queries are applied to them using the SPARQL language. This project was used to develop a system prototype to manage semantic master data (Lu et al., 2007).

Though a majority of this project was focused on how to reason against multiple ontologies, they did however implement a faceted search to find and view existing ontologies. The information these inquires would return are things such as a URI, textual description, list of classes, list of subject relations, and a list of object relations (Lu et al., 2007).

This method is an interesting example of an ontology repository because it uses database schemata to describe the metadata of an ontology. Furthermore it allows for the potential of using this data for deep reasoning on the ontologies stored within the system.

3.1.3 Lightweight Ontology Repository

The Lightweight Ontology Repository enables the sharing of ontologies between multi-agent systems (Pan et al., 2003), it was design so that ontology designers and agents can use this system with common web standards to publish and retrieve ontologies, along with associated metadata. The agents could access ontologies via a REST-style web service. With this project we see more emphasis put onto extra data associated with the ontology. Much like SOR, this repository keeps track of additional data via each ontology's metadata.

The metadata stored within this system is the same as the source *Description Frame Work Schema*. This ontology keeps track of information such as the ontology name, language, version and brief description, as well as information about the creator and how to contact them (Pan et al., 2003). This additional data gives a more complete understanding of the ontology being stored within the system, however it does not give a complete view. It is an abstracted view and further processing needs to be done to understand what the ontology is actually describing. The agents would need to have a pre-existing knowledge of which ontology they needed and could validate against the metadata to confirm that they received the correct ontology.

Though this system was designed with a specific domain in mind, it did emphasize the need to store and keep track of each ontology's metadata. However this system lacked search functionality due to the fact that the only way to receive ontologies was through request of exact ontology names. There is no notion of trust or quality within this system; the users or agent must assume that all ontologies stored within the system are valid and the information they are describing is accurate.

3.2 Proposed Method

As seen in previous sections many of these methods treat ontologies as files which are to be searched through. This is much like searching through source code for functionality. Unlike traditional markup languages like XML which only have flat data descriptions, ontologies have live and vibrant relations that may change once reasoned upon. In software engineering one of the key principles is that all elements of a package are modular and serve a specific task and thus these modules could be easily re-appropriated for different tasks. Since ontologies serve a function much like software this would lead to the conclusion that they too should be a self-contained entities, which are closed and consistent but still have facilities for extension (Maedche et al., 2003). This leads to the analogy of the software library.

This method would see ontologies grouped within packages, describing knowledge within a similar domain. Additionally these packages could be standardized and documentation could be developed to aid with distribution and understanding to the user we described in 2.3. Furthermore these packages or "Libraries" of ontologies could be standardized and controlled by experts of a specific domain to ensure quality, and consistency throughout all of the ontologies being distributed. This places a fundamental shift in responsibility of creation, production, and quality control to the creator of the ontology and removes substantial burden from the end user.

In this system, ontologies within any package would have to conform to a standard meta-ontology (described in detail in 3.2.1). This ontology would allow for detailed information about the syntactic and semantic entities stored within the ontology along with pertinent information about versioning, creation, controlling body, change information, intellectual property rights, as well as potentially add facilities to help stream line merging and mapping of external ontologies.

Finally, many software libraries have a review committee which gives rational to the inclusion or exclusion of functionality. This already has parallels in the ontology world; some of the most well known and most used ontologies are often controlled by a specific body and changes are released through them with rationales. For example, the Dublin Core and Friend of a Friend (FOAF).

3.2.1 The Meta-ontology

Much like the proposed use of a meta-ontology in (Pan et al., 2003), (Maedche et al., 2003) offers an alternative meta-ontology. (Maedche et al., 2003)'s ontology meta-ontology (OMO), describes key terms of the ontology, which project it has been used in, who the creator was and what are their credentials, the location of the ontology, etc.. This ontology is displayed in Figure 1.

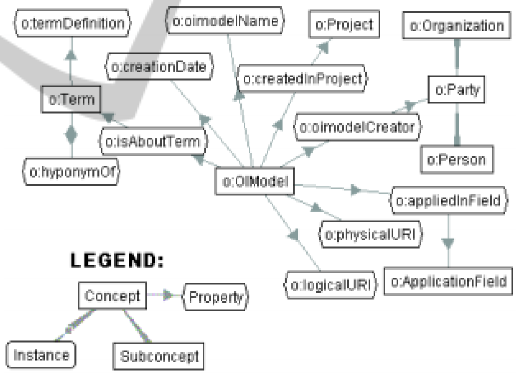

Figure 1: Ontology Meta-Ontology(OMO) (Maedche et al., 2003).

Though this ontology is relatively detailed it still lacks some key features necessary for use within our proposed library system. A notion of domain needs to be applied to this ontology, as to assert ontologies within the specific instance of the library belong to that domain. A notion of change and version is necessary as well (this is described in detail in section 3.2.2). For additional robustness and extendability it is necessary to add facilities to handle merging and mapping.

There are also social factors we can add to the meta-ontology to give a more complete description of the ontologies being stored. One such element is intellectual property and use rights of an ontology. This would be incredibly important in a business setting where you would not want clients to redistribute ontologies that contained important company data. Links to documentation could also could aid the user in further understanding of the rationale and relationship meaning of the ontology. The last aspect that could be added is the language in which the elements of the ontology are described, be it English, Spanish, Polish, Chinese, etc., and offer different translations of the ontology if they exist.

3.2.2 Version Information

Version information is very important when dealing with ontologies over time. Dependencies change, relations change, and ultimately the structure of the ontology will change. In the method being proposed change will be categorized as follows:

1. *Additions - when entities or relations are added to the ontology*
2. *Subtractions - when entities or relations are removed from the ontology*
3. *Modifications - when entities or relations are changed within the ontology*

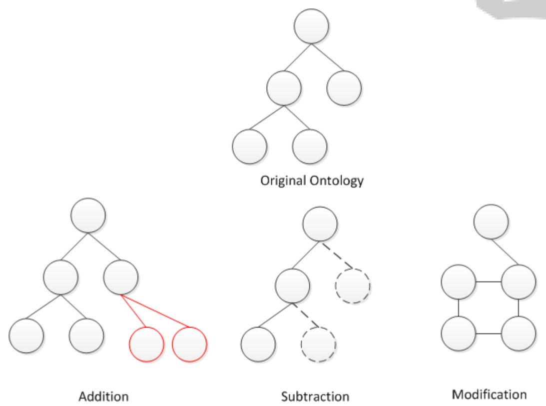

Figure 2: A visual representation of the types of changes which can be applied to an ontology.

Additions have the least impact on the overall structure of the ontology, since new information is being added to give a more complete description within the ontology and should have little to no effect on the existing relations. Subtraction can have a larger impact on the ontology as it is removing information from the ontology; this could affect the end user as they may have relied on this information in their application. Modifications have the most impact on the ontology and the largest ramifications to the user. Since the structure and relations change the reasoning and conclusions the ontology originally provided will change. This will affect an end user's application dramatically.

"It has been argued that ontology versioning, and, in particular, compatibility determination, cannot be performed automatically" (Heflin and Pan, 2004)(Flouris et al., 2007), and thus the necessity of integrating detailed notion of versioning within this library system to aid users on deciding whether or not to use the most current version of an ontology.

3.2.3 The Review Process

Many of the systems reviewed in this paper strictly use user-based rating systems to review or rate the effectiveness of an ontology. (Noy et al., 2009) heavily relays on the assumption that a satisfied user will post reviews on used ontologies, along with the information where the ontology was applied.

With a library model, a review will occur before a new version is released. This ensures that only ontologies of the upmost quality are released. Preferably the creators/maintainers of the library will review its contents, ensuring that it conforms to the library standards, as well as any modifications will not change focus or integrity of the ontologies being distributed.

This kind of release model can be seen through the C/C++ and Java standard libraries as well as the Linux kernel project.

3.2.4 Library over Repository

The method should be independent of domain as ontologies are ontologies, much like source code will be source code. One could make domain specific applications but essentially each application will be no different than the applications that proceeded it. Much like software libraries, functionality may be vastly different but distribution within packages is standardized. This makes deployment simple, which in turn makes descriptions managable, and ultimately simplifies the distribution process. The standardization makes the ontology make sense in any field.

Therefore, the idea of doing text search through complex structures is of limited value. Ontologies are complex functional objects, with properties and relations that need to be understood to comprehend what the ontology describes. Keyword searches often results in objects with the most of the entities the same but it may not have any relationship to what the ontology itself describes. While the repository relies heavily on keyword based search, the library being proposed uses a meta-ontology to link all ontologies within the library. This allows for reasoning based searches, which provide more accurate results than simple keyword based searches (Maedche et al., 2003).

The benefits of ontology libraries are standardization across all domains and a facility for deep meaning searches.

4 DISCUSSION & CONCLUSIONS

The ontology library proposed within this pape, is not just a technical implementation but more of new approach to this problem. Often when teams set out to create an ontology library/repository they have a set of features they wish to implement although they may not be considering the needs of the user. After implementation there is little literature examining its success. When designing a system with a library in mind there are several existing metrics we can apply to the library to see if it meets the criteria necessary to promote usability. With the aid of software metrics we can validate that the library system itself is flexible, reusable, extendable, and has met almost any needs of a possible user (Frakes and Terry, 1996).

Fundamentally the system being proposed is shifting the onus of evaluating, validating, and understanding the ontology to the system and creator. This is logical as the creators should have the best understanding of what the ontology is supposed to be representing. With this shift, comes standardization through the meta-ontology, giving much more context to what is contained in an ontology opposed to just its title and entity relations.

The meta-ontology proposed takes into consideration social aspects which are not necessarily considered, that of intellectual property and documentation. Systems like those seen in (Noy et al., 2009) and (Lu et al., 2007), offer statistics on the number of classes and relations and do not go into deeper description of their meanings.

Finally the review panel: the organization who produces an ontology library needs to take responsibility and control of all changes, and future iterations of the library for this method to work.

Though there exists many systems which try to offer a flavour of ontology repository, they often lack the depth and breadth need to have a true non-domain specific method to share, distribute, and discover ontologies.

ACKNOWLEDGEMENTS

The authors would like to thank the Guelph Ontology Team for help and support while developing these research ideas.

REFERENCES

Cardoso, J. and Sheth, A. (2005). Introduction to semantic web services and web process composition. In Cardoso, J. and Sheth, A., editors, *Semantic Web Services and Web Process Composition*, volume 3387 of *Lecture Notes in Computer Science*, pages 1–13. Springer Berlin / Heidelberg.

Ding, Y. and Fensel, D. (2001). Ontology library systems: The key to successful ontology re-use. In *Stanford University 2001; S*, pages 93–112.

Flouris, G., Manakanatas, D., Kondylakis, H., Plexousakis, D., and Antoniou, G. (2007). Ontology change: classification and survey.

Frakes, W. and Terry, C. (1996). Software reuse: metrics and models. *ACM Comput. Surv.*, 28(2):415–435.

Gillespie, M., Hlomani, H., Kotowski, D., and Stacey, D. (2011). A knowledge identification framework for the engineering of ontologies in system composition processes. In *Information Reuse and Integration (IRI), 2011 IEEE International Conference on*, pages 77 – 82.

Gruber, T. R. (1995). Toward principles for the design of ontologies used for knowledge sharing. *Int. J. Hum.-Comput. Stud.*, 43(5-6):907–928.

Guarino, N., Oberle, D., and Staab, S. (2009). What is an ontology? In Staab, S. and Rudi Studer, D., editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 1–17. Springer Berlin Heidelberg.

Heflin, J. and Pan, Z. (2004). A model theoretic semantics for ontology versioning. In McIlraith, S., Plexousakis, D., and van Harmelen, F., editors, *The Semantic Web ISWC 2004*, volume 3298 of *Lecture Notes in Computer Science*, pages 62–76. Springer Berlin / Heidelberg.

Lu, J., Ma, L., Zhang, L., Brunner, J.-S., Wang, C., Pan, Y., and Yu, Y. (2007). Sor: a practical system for ontology storage, reasoning and search. In *Proceedings of the 33rd international conference on Very large data bases*, VLDB '07, pages 1402–1405. VLDB Endowment.

Maedche, A., Motik, B., Stojanovic, L., Studer, R., and Volz, R. (2003). An infrastructure for searching, reusing and evolving distributed ontologies. In *In: Proceedings of WWW 2003*, pages 439–448. ACM Press.

Noy, N. F., Shah, N. H., Whetzel, P. L., Dai, B., Dorf, M., Griffith, N., Jonquet, C., Rubin, D. L., Storey, M.-A., Chute, C. G., and Musen, M. A. (2009). Bioportal: ontologies and integrated data resources at the click of a mouse. 37(suppl 2):W170–W173.

Pan, J., Cranefield, S., and Carter, D. (2003). A lightweight ontology repository. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, AAMAS '03, pages 632–638, New York, NY, USA. ACM.

Rubin, D. L., Moreira, D. A., Kanjamala, P. P., and Musen, M. A. (2007). Bioportal: A web portal to biomedical ontologies. In *Conference for theAdvancement of Artificial Intelligence (AAAI) 2008*.

Sure, Y., Staab, S., and Studer, R. (2009). Ontology engineering methodology. In Staab, S. and Rudi Studer, D., editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 135–152. Springer Berlin Heidelberg.

Uschold, M., Healy, M., Williamson, K., Clark, P., and Woods, S. (1998). Ontology reuse and application. In *Proceedings of the 1st International Conference on Formal Ontology in Information Systems (FOIS'98)*, pages 179–192. IOS Press.