Journal of Theoretical and Applied Information Technology
15th October 2012. Vol. 44 No.1
© 2005 - 2012 JATIT & LLS. All rights reserved·

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A WEB-BASED TOOLKIT FOR LARGE-SCALE ONTOLOGIES

[1]Yuxin Mao
[1] School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou 310018, P.R. China
E-mail: [1]maoyuxin@zjgsu.edu.cn

ABSTRACT

There is no doubt that those large-scale domain ontologies are playing a critical role in building a large variety of semantic-based systems. It's important and urgent to share and reuse large-scale ontologies to support semantic-based applications in a more efficient way. In this paper, we propose a web-based toolkit for building and reusing large-scale ontologies. The toolkit consists of a web-based ontology editor as well as a modulation-based API for ontology reuse. The web-based ontology editor supports cooperative online development of large-scale ontologies. It distinguishes itself from other editors by an easy-to-use interface. A modulation-based API is implemented for manipulating modulation from large-scale ontologies. It makes it possible to extract frequently-used portions from large-scale ontologies and cache those portions of knowledge in database for potential reuse. Moreover, we have evaluated the performance of the toolkit by simulation and application. In summary, the proposed toolkit is able to support building and reusing large-scale ontologies.

Keywords: *Ontology, Web-based, Sub-Ontology, Reuse, Reasoning*

1. INTRODUCTION

As the foundation of the Semantic Web [1], ontologies [2] are the specification of conceptualisations, used to help programs and humans share knowledge. Encoding domain-specific knowledge in terms of ontologies provides a possible approach to overcome the problem of semantic heterogeneity in modern information systems. Nowadays, ontologies are increasingly seen as a key technology for enabling semantics-driven knowledge processing [3]. To overcome the problem of semantic heterogeneity and encode domain knowledge in reusable format, large-scale ontologies will play a critical role in developing semantic-based (or knowledge-based) systems.

There is no doubt that those large-scale domain ontologies are playing a critical role in building a large variety of semantic-based systems that typically operate on and communicate with statements in some formal knowledge representation. However, including large-scale ontologies in their complete form could incur an unnecessarily high storage and maintenance cost. Therefore, to knowledge-intensive domains like bioinformatics or traditional Chinese medicine (TCM), it's important and urgent to share and reuse large-scale ontologies to support semantic-based applications in a more efficient way.

In this paper, we propose a web-based toolkit for building and reusing large-scale ontologies. The toolkit consists of two parts, a web-based ontology editor as well as a modulation-based API for ontology reuse. The remaining of the paper is organized as follows. In Section 2, we introduce a web-based ontology editor for modeling and building large-scale ontologies. In Section 3, we present a modulation-based API for reusing large-scale ontologies. In Sections 4 illustrate the performance evaluation of the toolkit. Section 5 concludes the paper with an outlook to future research directions.

2. WEB-BASED ONTOLOGY EDITOR

Recent advent of the Semantic Web has facilitated the incorporation of various large-scale ontologies in many disciplines especially biology and medicine, such as Gene Ontology (GO) [4] for gene product, UMLS [5] for integrating biomedical terminology, MGED Ontology [6] for microarray experiment and so on. Many ontology editors already exist and most of them are offline and general tools for constructing domain ontologies such as OilEdit [7], Protégé [8] and so on. As a most popular and powerful ontology editor, Protégé is a tool that allows users to construct an ontology. However, Protégé fails to satisfy some disciplines like biology or TCM with complex real-life requirements. Moreover, Protégé does not support web-based collaborative work well for ontology development. To overcome the problem of semantic heterogeneity and encode domain knowledge in reusable format, we need an integrated toolkit to develop large-scale ontologies.

Therefore, we develop a web-based ontology editor that allows users to edit and explore ontology online (see figure 1). The editor runs on the server-side and publishes large-scale ontologies to users through Web services. No special clients are required and users can browse and edit ontologies anywhere with their web browsers. It supports cooperative online ontology editing, incorporates a data-base back-end for large-scale ontology storage, and interacts with several popular ontology formats.

The editor also provides a tree-based view for classes and a form-based view for instances of an ontology. Figure 1 shows the standard user interface of the editor. The interface is divided into a set of panels—Figure 1 has three panels visible. The class panel (1) shows the class hierarchy, the instance panel (2) shows a list of instances, and the property panel (3) shows the details of a selected instance. The property panel consists of several slots, with each corresponding to a class property. To a specific instance, users can specify its property value by inputting either literals or related instances in a pop-up panel (4) from slots. Besides, we also implement some additional functions (5) like searching a class or instance in the scope of the ontology, counting the number of classes or instances and so on, to facilitate development.

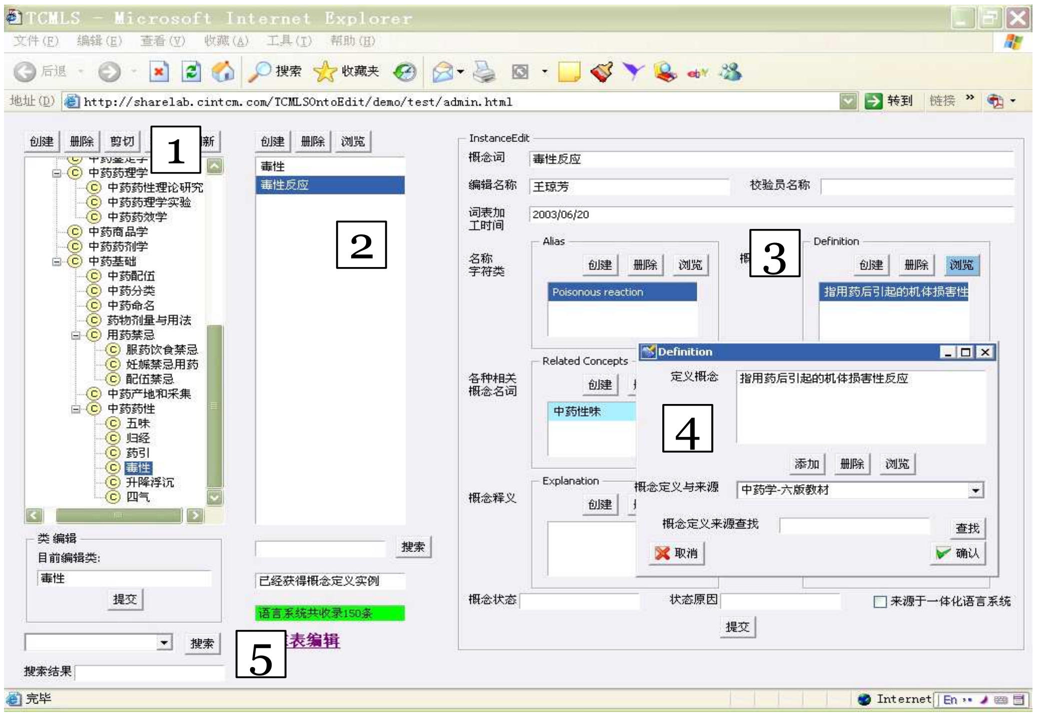
We employ a layered privilege model for the ontology editor. Users that play different roles in the process of ontology development own different privileges. Assume the target ontology is divided into several categories (e.g. disease, biomedicine, acupuncture and so on, to a TCM ontology), then there are mainly four kinds of privileges: reader, developer, checker and administrator.

(1) Readers can browse all the contents of the ontology.

(2) Developers can input, modify and delete instances within a category but have no privilege to manipulate the classes of the category.

(3) Checkers own the privilege to manipulate both classes and instances in a category.

(4) Administrators have the global privilege to all categories of the whole ontology.



http://sharelab.cintcm.com/TCMLSOntoEdit/demo/test/admin.html

目前编辑类: 毒性

毒性
毒性反应

已经获得概念定义实例
语言系统共收录150条

概念词: 毒性反应
编辑名称: 王琼芳
词表加工时间: 2003/06/20
Alias: Poisonous reaction
Definition: 指用药后引起的机体损害性
Related Concepts: 中药性味
定义概念: 指用药后引起的机体损害性反应
概念定义与来源: 中药学-六版教材

Fig. 1 The default user interface for the ontology editor. Note that the ontology is currently written in Chinese. In this figure, the class toxicity is selected by user, which is a sub-class of herbal medicine property. In the instance panel, the instance toxic reaction is selected. Toxic reaction has a property definition, whose value is another instance, which is shown in the pop-up panel.

The classic Web application model adopts a "click, wait & re-fresh" user interaction and a synchronous request/response communication mechanism, which results in slow, low productivity and inefficient Web applications. When an ontology grows in size, it's intolerable that users wait several minutes for page refresh in online development. We provide users with a graphical inter-face based on AJAX (Asynchronous JavaScript And XML), which is a combination of techniques for submitting server requests in asynchronous communication and returning data from the server to the user without the necessity of waiting for a page load. For example, an ontology developer wants to add a sub-class called haematology to the existing class biochemistry in a TCM ontology. He selects the class biochemistry in the class hierarchy tree, add a class under biochemistry and input the class name, haematology. Things are finished in a single page (see figure 1). The user sees the impact of his operations immediately without additional sub-mission or waiting, just as typing a document on his PC.

The web-based ontology editor for developing large-scale ontologies distinguishes itself from other editors by providing an AJAX-based graphical interface, cooperative online development and layered privilege control.

3. ONTOLOGY REUSE BY MODULATION

Taking into account the locality of knowledge reference, we propose to represent those context-specific portions of knowledge from the whole ontology as sub-ontologies (SubO) [9]. A large-scale ontology like GO contains relatively complete knowledge about the domain of interest it focuses on. Our conjecture is that the activities of a semantic-based system are always local to a subset of known information (e.g., a large-scale domain ontology). Taking into account the locality of knowledge reference, we propose to represent those context-specific portions of knowledge from the whole ontology as SubOs, which can be reused in semantic-based systems. We give a formal definition of SubO based on the semantic structure of ontology as follows:

Sub-Ontology. Formally, a sub-ontology (SubO for short) can be represented as a triple <*C*, *K*, *O*>, where *C* is a set of contextual concepts to feature the SubO as the context of problem-solving to the source ontology, *K* is a self-contained knowledge set relevant to *C*, and *O* is a link to the source ontology.

The concept (note that in the rest of this paper, we just take *concept* and *class* as the same term to ontology) set *C* is used to capture the features of knowledge reference and elements in *C* are the concepts related with a problem-solving scenario at most. Contextual concepts can vary with contexts and it makes sense that a SubO can be reused in different contexts. The self-contained characteristic of *K* means that all knowledge references are involved in *K*.

For example, the family 'phosphofructokinase' in InterPro (Mulder et al, 2003) is annotated to GO:0006096 (glycolysis), GO:0003872 (6-phosphofructokinase activity), and GO:0005945 (6-phosphofructokinase complex). Then the SubO for these three GO terms is shown in figure 1: the concept set is {GO:0006096, GO:0003872, GO:0005945} and the knowledge set contains all the GO terms in the figure. Therefore, ontology reuse for GO-based systems can be achieved in terms of utilizing SubOs. In other words, SubO is treated as the basic unit of ontology reuse.

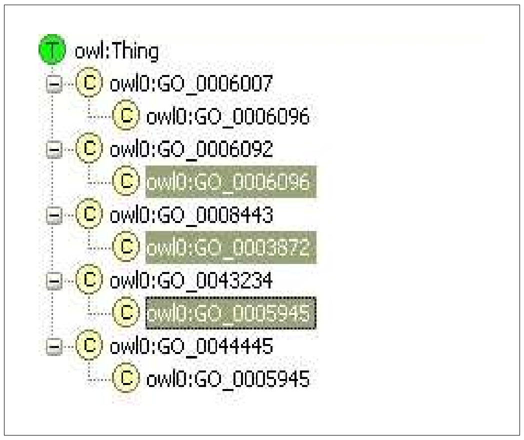


Fig. 2 The Subo For Annotating The Family 'Phosphofructokinase' In Interpro

We implement a modulation-based API for manipulate SubOs from large-scale ontologies. The API is an open source toolkit for manipulating SubOs from large-scale ontologies. We can extract, store and query SubOs efficiently with the API to achieve ontology reuse. Ontology users or semantic-based systems can use these APIs to extract, store or query SubOs from large-scale ontologies.

Given a set of terms *C*, SubOs potentially required for semantic-based systems can be directly extracted from ontologies by our API. The process of extraction is reduced into a group of traversals. Briefly, the extraction of SubOs proceeds as follows:

(1) Class labels if any (e.g. transcription, DNA-dependent) in *C* are converted into class IDs (e.g. GO_0006350).

(2) Starting at a class *c*, we traverse classes related to *c* through relation links (e.g. is-a or part-of) in the ontology.

(3) The traversal is performed with the breadth-first algorithm. The breadth-first traversal terminates when a depth of n has been met or there are no more terms to be traversed.

(4) For each input term in *C*, we find the corresponding ontology class, perform a traversal in the ontology and get a subset of ontology.

(5) Finally we combine all subsets together and the final set is just the knowledge set *K* of the required SubO.

The traversal depth n is an alterable parameter to the API. Note that the terminal classes in a traversal path are called edge classes whose definitions are not allocated in *K*. For example, starting from GO_0006349 and GO_0006355, we traverse GO until a depth of 2 has been met. The classes (e.g. GO_0050789) that terminate the traversal are edge classes. The final SubO is shown in figure 3, and the gray nodes denote edge classes. Except edge classes, every class in the SubO has its complete definition in *K*.

Newly extracted SubOs from large-scale ontologies in terms of <*C*, *K*, *O*> can be directly used as a Jena [10] ontology model or stored in a repository to be reused in the future. Next time any request with a similar set of ontology terms is transferred to the repository of SubOs rather than directly to the ontology. The API supports database storage for SubOs and query over database by concept sets. Given a set of class IDs or class labels, the API queries database and returns the best-matched SubO.

In relational database, the three elements of SubO are stored in different fields respectively. Moreover, an additional field called frequency is reserved to record how often a SubO is reused by semantic-based systems. The frequency field is useful to systems that require dynamic evolution on SubOs. This calls for the additional capability of semantic-based systems to keep evolving (modify, augment, prune, etc.) frequently used SubOs, emerged to be specialized in their own areas of concern [9]. In this way, a repository of SubOs is used as a cache for large-scale ontologies. Caches work well because of a principle known as locality of reference. By keeping as much of the data as possible in the SubO repository, semantic-based systems avoids accessing the whole ontology.

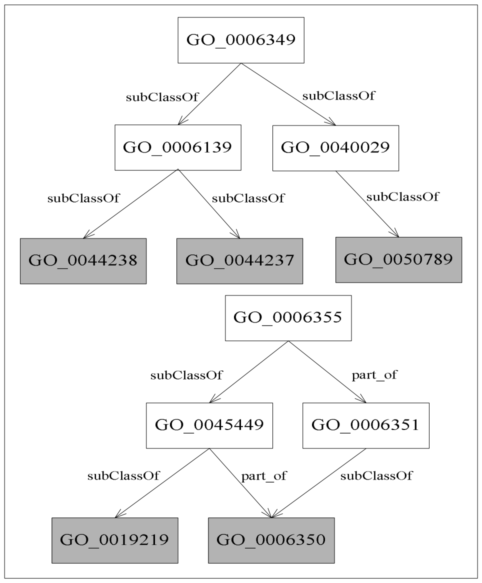


Fig. 3 A Subo With A Traversal Depth 2.

4. PERFORMANCE EVALUATION

In order to evaluate the performance of the proposed toolkit, we use it to develop a large-scale ontology for TCM called Traditional Chinese Medicine Language System (TCMLS) [11]. By using the web-based toolkit, we have developed the TCM ontology (see the statistics in table 1). As we can see in the table, the toolkit is able to support large-scale ontology. The ontology has become large enough to cover different aspects of the TCM discipline and is used to support semantic-based systems for TCM.

Table 1 *The Major Result Of The TCM Ontology By The Toolkit.*

Item	Value
Response Time (ms)	500
Query Response Time (ms)	1000
Number of Classes	10370
Number of Instances	78754

Moreover, we have evaluated the modulation-based API of the toolkit. The most important and frequent operation for ontology reuse is SubO extraction. Therefore we investigate the efficiency of SubO extraction by using the toolkit. We randomly extract about 100 SubOs from the TCM ontology with different depth values and record the average extraction time for each of them. Table 2 just illustrates the time of extracting a SubO under different depth values. From table 2 we can see that the time is increased to a large extent when the depth value grows from 1 to 5. It makes sense that the extraction process is more complex under a larger depth value. The toolkit is able to support extracting SubOs from large-scale ontologies under certain time.

Table 2 The Time Of Extracting A Subo Under Different Depth Values By Using The Toolkit.

Depth	1	2	3	5
Time (10^5 ms)	3.2104	3.9726	4.7403	5.2203

Moreover, we have also evaluated the cache performance of the API. The primary metrics used to evaluate the cache performance are hit-ratio and response time. A cache with higher hit-ratio and lower response time is better. We perform a number of operations to the cache with SubOs from the TCM ontology. We calculate the average hit-ratio and the response time for a number of operations (e.g. 500).

Table 3 The Cache Performance Of The Toolkit.

Number	500	1000	2000
Hit-ratio	30%	45%	53%
Response Time (ms)	100	80	30

From table 3, we can see that the performance of the cache is improved when the number of operations increases. The cache is able to improve the performance of semantic-based systems by keeping frequently-used SubOs in a repository.

5. CONCLUSION

Building and reusing large-scale ontologies is important to implement semantic-based systems. In this paper, we propose a web-based toolkit in order to achieve the purpose. The toolkit consists of a web-based ontology editor as well as a modulation-based API for ontology reuse. The web-based ontology editor supports cooperative online development of large-scale ontologies. It distinguishes itself from other editors by an easy-to-use interface. A modulation-based API is implemented for manipulating SubOs from large-scale ontologies. It makes it possible to extract frequently-used portions from large-scale ontologies and cache those portions of knowledge as SubOs in database for potential reuse. In summary, the proposed toolkit is able to support large-scale ontologies. Future works include how to extend the toolkit to support more ontology languages and how to integrate a reasoning engine to support ontology reasoning.

ACKNOWLEDGEMENTS

This work is partially supported by a grant from a NSFC Program (NO. NSFC61003309) and Science and Technology Department of Zhejiang Province Program (NO. 2010C33045, and 2010C33047).

REFRENCES:

[1] T. Berners-Lee, Hendler J, Lassila O. “The Semantic Web”, *Scientific American*, Vol. 284, No. 5, 2001, pp. 34-43.

[2] T. Gruber, “A Translation Approach to Portable Ontology Specifications”, *Knowledge Acquisition*, Vol. 5, No. 2, 1993, pp. 199-220.

[3] A. Maedche, B. Motik, L. Stojanovic, R. Studer, and R. Volz, “Ontologies for Enterprise Knowledge Management,” *IEEE Intelligent Systems*, Vol. 18, No. 2, 2003, pp. 26-33.

[4] M. Ashburner, C.A. Ball, J.A. Blake, et al., “Gene Ontology: tool for the unification of biology”, The Gene Ontology Consortium. *Nat. Genet.*, Vol. 25, 2000, pp. 25-29.

[5] O. Bodenreider, “Unified medical language system (umls): integrating biomedical terminology”, *Nucleic Acids Research*, Vol. 32, No. D, 2004, pp. D267-D270.

[6] L. Patricia, et al., “The MGED Ontology: a resource for semantics-based description of microarray experiments”, *Bioinformatics*, Vol. 22, No. 7, 2006, 866-873.

[7] S. Bechhofer, I. Horrocks, C. Goble, and R. Stevens, “OilEd: a Reason-able Ontology Editor for the Semantic Web”, *Proc. KI2001*, Vol. 2174, 2001, pp. 396-408.

[8] N.F. Noy, M. Sintek, S. Decker, et al., “Creating Semantic Web Contents with Protege-2000”, *IEEE Intelligent Systems*, Vol. 16, No. 2, 2001, pp. 60-71.

[9] Y. Mao, W.K. Cheung, Z. Wu, J. Liu, “Dynamic Sub-Ontology Evolution for Collaborative Problem-Solving”, *Proc. AAAI Fall Symposium*, v FS-05-01, 2005, pp. 1-8.

[10] B. McBride, "Jena: A semantic web toolkit", *Internet Computing*, IEEE, Vol. 6, Issue 6, 2002, pp. 55-59.

[11] X. Zhou, Z. Wu, A. Yin, et al., "Ontology Development for Unified Traditional Chinese Medical Language System", *Journal of Artificial Intelligence in Medicine*, Vol. 32, No. 1, 2004, pp. 15-27.