온톨로지 엔진의 유지, 관리를 위한 체인지 로거

아사드 마소드 가탁, La The Vinh, 이승룡, 구교호
경희대학교
e-mail : {asad.masood, vinhlt, sylee}@oslab.khu.ac.kr, yklee@khu.ac.kr

Change Logger: Towards Ontology Maintenance

Asad Masood Khattak, La The Vinh, Sungyoung Lee, Young-Koo Lee
Dept. of Computer Engineering, Kyung Hee University, Korea

Abstract

To accommodate constantly growing knowledge in scientific discourse that is revised over time by domain experts, we need to also evolve our ontology. The body of knowledge will get structured and refined as we develop a deeper understanding of issues. Keeping trail of new changes in semantically rich and formally sound mechanism has pragmatic advantages for providing the undo and redo facility and ontology recovery to a previous state. In this research, we have proposed a framework that support change logging and then using these logged changes for reverting ontology to a previous consistent state and visualization of change effects on ontology. The system is compared with ChangesTab of Protégé and the results depict better accuracy for our system.

1. Introduction

Ontologies are formal description of shared conceptualization of a domain of discourse. Ontology change management is the solution to the problem of deciding the modifications to perform in ontology in response to a certain need for change [1]. Ontology change management is a complicated and multifaceted task, which has led to the emergence of several different, but closely related, research areas. Ontology Integration, Merging, Versioning, and Evolution deal with different aspects of this problem [1]. Changes do occur in ontology and are reflected in the ontology by implementing these changes. As a result it evolves to a new state [2, 3]. Consequently, an ontology change management solution has to answer a number of questions [4] like; systems' overall working, "how to maintain all the changes in a consistent and coherent manner?" While other questions revolve around the applications of all these logged changes for the purpose of ontology recovery and visualization of change effects.

The goal of this research article is to provide preliminary experimental results for our semantic structure and framework [4, 5] for temporal traceability in ontology evolution management. We developed Change History Ontology (CHO) [4] for maintaining ontology changes semantically. We envisioned a number of applications for the logged changes such as, ontology change management, change in semantics of the concepts, ontology recovery in case the system crashes, query reformulation, reconciliation of ontology mappings, change traceability, and to some extent navigation and visualization of the changes and change effects [5]. We have implemented and build a framework as a plug-in for Protégé (an ontology editor) as '*Change Tracer*'. It automatically detects and logs all the changes happened to ontology using CHO, triggered by the change request from ontology engineer. After that, whenever required, the CHL changes are accessed. The plug-in roll-back and roll-forward any changes and get the ontology in any previous consistent state. We have compared our system results for change capturing with *ChangesTab* of Protégé and our system has outperformed *ChangesTab*.

2. System Implementation and Results

We envisioned our proposed framework (see Figure 1) as an enabling component for ontology editors. It doesn't provide ontology editing services. The framework is designed to be used as a plug-in for different ontology editors provided they support the hooks we have implemented. Different individual components in the framework have their own tasks, related to change history management. Change Logger component, for instance is responsible to preserve the changes using CHO (shown in Figure 2).

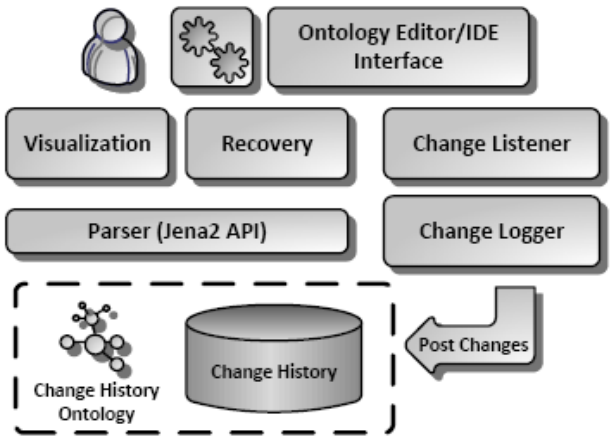

Figure 1. System architecture for Change Tracer

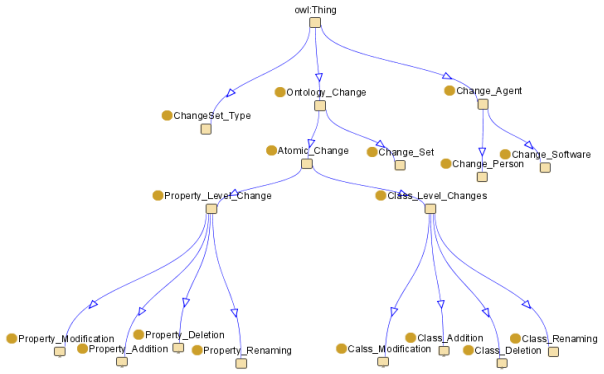

Figure 2. Change History Ontology (CHO)

Different modules of the framework have their own different task to perform; 1) Change Capturing is responsible for capturing different changes occurring to ontology during the life span of ontology. To capture these changes we have implemented 7 different java interfaces available in Protégé, Protégé-OWL, and Jena API's. 2) Change Logger logs all the captured changes in the CHL. 3) CHL is a repository that keeps track of all the changes and it uses the structure provided by CHO [4]. 4) Parser is a module that parses ontology changes based on user request. This parsing can be for any purpose like; logging, recovery, and visualization. 5) Recovery module is responsible for recovering ontology from one consistent state to another. This recovery is both i.e. roll-back and roll-forward recovery. In CHO we introduced the concept of *ChangeSet* for bunch of changes, so with the help of *ChangeSet* we can recover ontology properly and to any of its previous state. 6) Visualization module is responsible for visualization of ontology, ontology changes, and visual navigation of ontology changes. This help in proper and better understanding of ontology evolution behavior. We have extended the *TouchGraph* API for graph drawing in order to visualize the graph view of the ontology structure. Resources, such as concepts, are depicted as nodes. These nodes are connected through slots which are depicted as the edges in the visualization.

To validate the working of the proposed framework, we have developed a *TabWidget* plug-in, Change Tracer Tab, for Protégé ontology editor. The details of all the five main modules and their implementations are available in [5]. Here we provide the results analysis of our system against *ChangesTab* of Protégé.

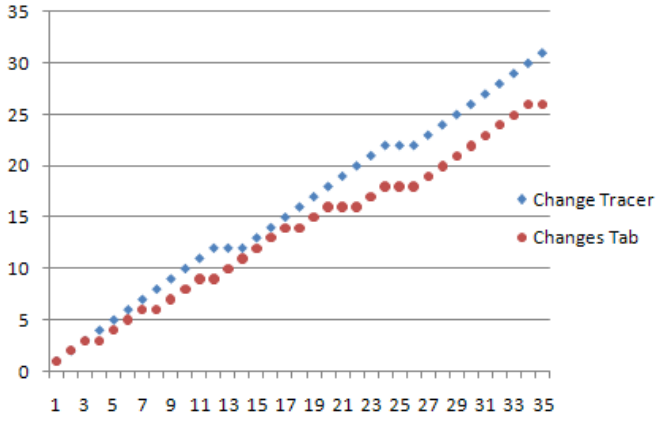

Figure 3, Comparison of *Change Tracer* against *ChangesTab* of Protégé

We have configured our system and *ChangesTab* in Protégé. Total of 35 different changes were made to *Documentation* ontology. Out of these 35 changes, *ChangesTab* captured 26 changes while our system captured 31 different changes. The x-axis in Figure 3 represent no of changes applied and y-axis represent captured changes.

We have also envisioned different applications of the stored ontology changes. The changes logged can be used for applications such as; 1) Ontology Recovery: is the application we discussed in this paper. 2) Query Reformulation: when ontology evolves from one state to another then the query posed on previous ontology versions are not valid for the new evolved one. To reformulate the query over evolve version we can use the stored changes. 3) Reengineering Ontology Mappings: when there are mappings available between two ontologies and one evolves from one state to another then the mapping are no more reliable. So the logged changes can be used for efficient reestablishment of broken mappings between ontologies. 4) Temporal Traceability: in this, we have all the ontology changes traced and stored in a repository. We can use these changes for predicting some temporal patterns about the change history. Also these can help for visual navigation of ontology history. These changes can also be used for future change prediction in different circumstances. 5) Managing Ontology Changes: this helps in properly managing ontology changes during evolution. Using the changes stored in CHL can help user understand the evolution/growth of ontology in different phases and the user can get complete understanding of the ontology in focus. Annotation can also be added to the changes to explain the reason for change, author of change, and timestamp of ontology change and this can help in minimizing the evolution effects on the dependent data, applications, and ontologies. This change management can also help in understanding the semantics of changes.

Acknowledgement

This research was supported by the MKE (Ministry of knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the NIPA(National IT Industry Promotion Agency)" (NIPA-2009-(C1090-0902-0002)).
Also, it was supported by the IT R&D program of MKE/KEIT, [10032105, Development of Realistic Multiverse Game Engine Technology].

References

[1] G. Flouris, D. Plexousakis, and G. Antoniou, "A Classification of Ontology Changes", In Semantic Web Applications and Perspectives, PISA, Italy, 2006.
[2] L. Stojanovic, A. Madche, B. Motik,, "User-driven ontology evolution management," In European Conference on Knowledge Engineering and Management, pp. 285-300, 2002.
[3] S. Castano, A. Ferrara, G. Hess, "Discovery-Driven Ontology Evolution". The Semantic Web Applications and Perspectives, PISA, Italy, 18-20 December, 2006.
[4] A. M. Khattak, K. Latif, S. Khan, N. Ahmed, "Managing Change History in Web Ontologies," Semantics, Knowledge and Grid, International Conference on, pp. 347-350, 2008.
[5] A. M. Khattak, K. Latif, S. Khan, and N. Ahmed, "Ontology Recovery and Visualization," Next Generation Web Services Practices, pp. 90-96, 2008.