Ideas for Collaborative Ontology Development on the Upcoming Web 3.0 Era

Matthias Loskyll[1] and Dominikus Heckmann[2]

[1] Saarland University, 66123 Saarbrücken, Germany, matthias@xantippe.cs.uni-sb.de
[2] DFKI GmbH, 66123 Saarbrücken, Germany, heckmann@dfki.de

Abstract. Ontologies continuously become larger and more complex, and therefore more and more difficult to maintain, to edit and to develop by one single person or a small group of experts. The basic principle of Web 2.0, on the other hand, is to use the willingness and knowledge of a huge community of users to create rich user-generated content. The obvious idea that comes to mind is to combine the technologies of the Semantic Web with the trend of the Web 2.0. In this paper we present UbisEditor, an easy-to-use web tool for the creation and manipulation of structured collective knowledge represented as ontologies. This web ontology editor is realized as part of the UbisWorld project (`http://www.ubisworld.org`) and already supports ontology editing techniques like adding new concepts, renaming and deleting, but also the creation of personalized ontology views.

1 Introduction

The characterization of Web 3.0 used in this paper is based on [11], in which Wahlster and Dengel define Web 3.0 as the integration of Semantic Web technologies with the principles of Web 2.0. Our approach perfectly fits to this definition of Web 3.0 because it combines ontology development, which forms the backbone of Semantic Web, together with the community approach of Web 2.0.
With the increasing importance of ontologies, several **ontology editors** have been developed. Two of the currently most popular ones are Protégé [6] and Swoop [2]. However, there are only few online editors for ontologies available. Recently, a web-based version of Protégé called WebProtégé [10] was released. An evaluation of different tools for collaborative ontology development during the CKC challenge [5] showed that users prefer Web interfaces for editing tools. Our prototype called UbisEditor uses Ajax technologies to load and send required data on demand such that it provides a lightweight and efficient way of performing ontology editing on the Web. Other differences of our approach compared with related work are: our ontology is stored in a database structure, editing can be performed using a context menu, we already have a role-based rights management, and we provide multilingual renaming of concepts.
Ontology segmentation or the creation of ontology views is an important issue. Since many ontologies constantly grow in size and are therefore more and more difficult to understand, maintain and edit, ontology segmentation shall help to extract only those parts of a large ontology that are relevant for a certain scenario. Seidenberg and Rector compare and evaluate several algorithms for extracting relevant segments in [8]. The key approach is to let the user select some relevant concepts and to automatically add superclasses, subclasses and other related concepts (e.g. classes contained in axioms and restrictions) to the ontology view. We implemented a similar, lightweight online approach as part of the UbisEditor.

2 The Web Ontology Editor

The idea behind UbisEditor is the following (see figure 1 left): Our basic ontology is created and edited by an ontology expert, who can use Protégé to perform this task. Our ontology community, on the other hand, extends the basic ontology via UbisEditor with the possible issue of dublicated and conflicting concepts. Another feature of UbisEditor is to allow an individual ontology user to create a personalized ontology segment.
The **UbisEditor** itself is part of the **UbisWorld** project, which focuses its research on ubiquitous user modeling [1] and Web 3.0. The latest tool set version UbisWorld 3.0 integrates Web 2.0-like services and Semantic Web technologies. The knowledge base of the system is built out of two ontologies:

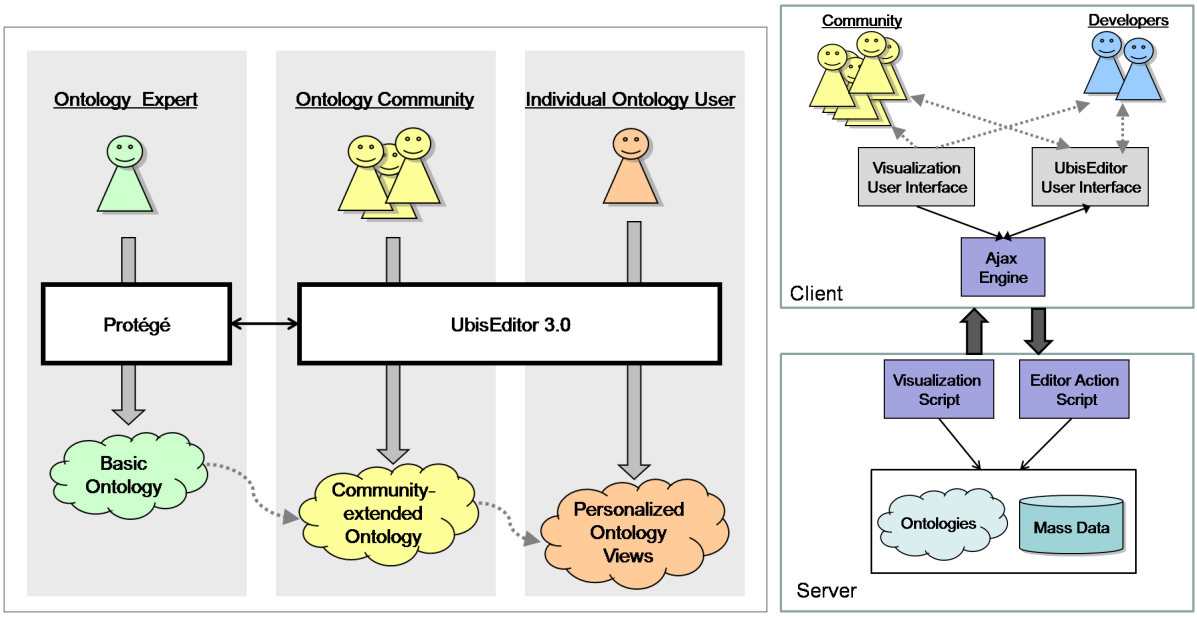

Fig. 1. Left: Collaborative Ontology Engineering in the Web 3.0 era with UbisEditor; Right: Abstract architecture of the UbisEditor and visualization system

GUMO (General User Modeling Ontology) and UbisOntology. These ontologies are represented as foldable trees whose nodes consist of classes and instances. In addition, several external ontologies (e.g. SUMO, DOLCE, OpenCyc) have been parsed and integrated into the UbisWorld. At the moment, further OWL ontologies can only be imported by our team, but we plan to provide a functionality to allow users to import and edit their own ontologies anytime soon.
All the trees can be browsed in a very efficient way by using Ajax technologies [4]. This means that the data needed to display the layer of a tree is sent to the client not before the user has opened the corresponding parent node. Using this visualization technique we are basically able to display arbitrary large trees. Another strength of our visualization technique is the clear structure achieved by displaying limitation nodes, which subsume large numbers of nodes in ten thousand or hundred packets, respectively.
UbisEditor uses the same graphical user interface as the UbisWorld ontology browser, but with additional features enabled like check boxes, a context menu, a drag&drop functionality and editable tree node labels.

2.1 Architecture of UbisEditor

Figure 1 (right) shows the basic architecture of the UbisEditor and the ontology browser. Both the developers of our group and our user community can use the UbisEditor to manipulate the different ontologies. When committing changes, a script stores the appropriate data in the database and a script for building the ontology trees sends the updated information back to the client. With this technology, we are able to overcome the performance issues that appeared in the context of web ontology editors so far.

2.2 Editing Process and Quality Control

The UbisEditor already supports the most important functionalities for editing an ontology: creating new classes or instances, renaming and deleting objects as well as changing a node's parent via drag&drop. By using a context menu, we provide an efficient and easy-to-use way of performing these editor actions. This context menu is adaptive to the rights of the user, which depend on a role-based rights management. This means that only users belonging to a certain role can delete objects, for example. In addition, we define subbranches that are only visible to special groups of users.

When the editing process is finished, the user has to push a button to commit the performed changes to the server. Then a script executes the actions corresponding to the type of the changes. By changing only those values in the database that really have changed, the updated tree can be reloaded and displayed immediately to the user.
For each editing process performed with UbisEditor we store the identifier of the registered user who is responsible for the appropriate changes. By doing so, we are able to find out whether a user constantly inserts low quality content. In addition, UbisWorld already has a five-star rating system available, with which our user community can help to ensure the quality and integrity of our content [7]. A possible extension could be to make it possible to rate the raters, too.

2.3 Multilingual Editing of Labels

UbisLabel is our approach to define multilingual labels in identifiers for semantic web and ubiquitous computing. We use UbisLabels to denote concepts in our ontologies. The basic idea is to use only ASCII characters in names, even if selected special characters are allowed. The syntax of the UbisLabel approach is described below.

```
[D]   UbisLabel     ::= SimpleLabel | LanguageLabel | MultiLabel
[D]   SimpleLabel   ::= UbisToken
[D]   LanguageLabel ::= (LanguageCode)+ ".." UbisToken
[D]   MultiLabel    ::= UbisLabel ("..." UbisLabel)+
[D]   LanguageCode  ::= ISO_639_1 | "ME"
[D]   ISO_639_1     ::= "EN" | "DE" | "IT" | "JP" | "YI" | ...
```

So this approach not only enables multiple labels for any concept from the knowledge representation point of view, but also allows the adaptive selection and presentation of a label according to a situation-aware strategy.
In order to make a multilingual labeling of the concepts of the different ontologies possible, we provide an additional method to edit the label of an object. When selecting a node of an ontology tree, a grid appears on the right-hand side of the UbisWorld web page. This grid has two columns (Language and Label) where the different labels that already exist for this concept are shown together with the corresponding language. Further translations of the concept's name can be added by first selecting a language out of a combo box containing the different iso 639 language codes and entering the appropriate name into a text editor field afterwards. This feature facilitates a collective generation of multilingual ontology concepts.

2.4 Aspects of Collaborative Ontology Development

The idea to facilitate a collaborative ontology editing raises several issues. One important task is the implementation of a transaction management to avoid problems when multiple users edit the ontologies and commit their changes simultaneously. However, since our system stores the different ontologies in a database back end, the concurrency control will probably be solved on the database level. In addition, we think about implementing a locking mechanism on the subtree or ontology module level to avoid inconsistencies.
When different members of a community collectively extend an ontology, different opinions inevitably can cause conflicts to occur. Consequently, a conflicts resolution is an essential functionality. We plan to solve this problem using a combined approach consisting of a community-based and a role-based conflicts resolution strategy similar to the one proposed by Li et al. in [3].
In order to avoid dublicated concepts, we already implemented a search engine for the UbisWorld system, which can be used to find classes, instances and properties of the UbisOntology. This is a very important feature with regard to the editing process using UbisEditor. When a user plans to insert a new element into the UbisWorld ontology, but is not sure whether an appropriate concept already exists, the UbisSearch can be used to remove ambiguity.
In the near future we plan to develop a change management system for UbisEditor and to extend the rating system such that performed ontology editing operations can be rated. By doing so, the change history and versioning information of each ontology concept can be displayed on the right side of the website helping the user to understand the evolution of the ontology. Our user community can help do discuss and decide whether performed changes to the ontology should be kept or reverted. In this

context, the availability of tagging and annotation facilities as supported by Collaborative Protégé [9] becomes essential. The UbisWorld already provides a simple tagging functionality, but more advanced features for annotating and discussing ontology concepts and changes are needed.

2.5 Personalized Ontology Views

Another important feature supported by UbisEditor is the possibility to create ontology views, i.e. to extract only those parts of the ontology that are relevant for the scenario on hand. In editor-mode, the ontology trees are displayed with a check box on the left-hand side of each node. So the user can easily select the needed concepts and create a personalized ontology view by pushing the appropriate button. Then an OWL-file is provided for download. We always integrate the direct and indirect superclasses of the selected concepts traversing the hierarchy tree upwards until the top node of the ontology. In addition, we let the user decide whether only the selected concepts (and the appropriate superclasses) shall be included in the ontology view or also their subclasses. At the moment, the UbisWorld does not provide functionalities to define and display ontological restrictions and axioms. Consequently, we do not consider classes contained in these definitions for the creation of ontology segments so far. However, we plan to do so in the near future.

3 Conclusions and Future Work

We have presented UbisEditor, a Web tool which provides lightweight functionalities for efficient collaborative ontology editing using an Ajax-based visualization technique. The central idea of our approach is to facilitate the distributed extension of our basic ontologies, that were created by experts, performed by our ontology community.
UbisEditor is part of the UbisWorld 3.0 tool set that can be tested online at www.ubisworld.org. Our initial research results are currently transferred to the domain of distributed technology enhanced learning in the EU project GRAPPLE.
In the near future, we plan to extend the functionality of UbisEditor in order to support the definition of ontological axioms, the creation of properties and the user-driven merging of ontologies. Concerning the support of collaborative ontology development, mechanisms for concurrency control, conflicts resolution and change management are essential. Additionally, we are going to perform an extensive evaluation of the editor features and of our ontology visualization techniques.

References

1. D. Heckmann. *Ubiquitous User Modeling.* Berlin: Akademische Verlagsgesellschaft Aka GmbH, 2006.
2. A. Kalyanpur, B. Parsia, E. Sirin, B. C. Grau, and J. Hendler. Swoop: A web ontology editing browser. *Web Semantics: Science, Services and Agents on the World Wide Web*, 4(2):144–153, June 2006.
3. M. Li, D. Wang, X. Du, and S. Wang. Ontology construction for semantic web: A role-based collaborative development method. pages 609–619. 2005.
4. M. Loskyll. Ontological and Ajax-based Extension of UbisWorld. Bachelor thesis, 2007. Chair for Artifical Intelligence, Prof. Dr. Dr. h.c. mult. Wahlster, Saarland University.
5. N. F. Noy, A. Chugh, and H. Alani. The ckc challenge: Exploring tools for collaborative knowledge construction. *Intelligent Systems*, 23(1):64–68, 2008.
6. N. F. Noy, M. Crubezy, R. W. Fergerson, H. Knublauch, S. W. Tu, J. Vendetti, and M. A. Musen. Protégé-2000: an open-source ontology-development and knowledge-acquisition environment. *AMIA ... Annual Symposium proceedings / AMIA Symposium. AMIA Symposium*, 2003.
7. P. Recktenwald. A Web 2.0 Rating System for UbisWorld. Bachelor thesis, 2007. Chair for Artifical Intelligence, Prof. Dr. Dr. h.c. mult. Wahlster, Saarland University.
8. J. Seidenberg and A. Rector. Web Ontology Segmentation: Analysis, Classification and Use, 2006.
9. T. Tudorache, N. Noy, S. Tu, and M. Musen. Supporting collaborative ontology development in protégé. pages 17–32. 2008.
10. T. Tudorache, J. Vendetti, and N. F. Noy. Web-protege: A lightweight owl ontology editor for the web. In C. Dolbear, A. Ruttenberg, and U. Sattler, editors, *OWLED*, volume 432 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.
11. W. Wahlster and A. Dengel. Web 3.0: Convergence of Web 2.0 and the Semantic Web. *Telekom Technology Radar II, Juni*, pages 1–23, 2006.