Co-Protégé: Collaborative Ontology Building with Divergences

Alicia Díaz, Guillermo Baldo
Lifia, Fac. Informática- UNLP, CC 11, 1900
La Plata, Argentina
[alicia, gbaldo]@sol.info.unlp.edu.ar

Gérôme Canals
Loria, Campus Scientifique, B.P. 239,
54506 Vandœuvre-lès-Nancy cedex, France
gerome.canals@loria.fr

Abstract

In this paper we present an innovative approach to develop a domain ontology in collaborative fashion. This approach is synthesized in a groupware application which is called Co-Protégé, a set of plug-ins which extends Protégé. This approach is innovative because Co-Protégé enables the coexistence of divergent conceptualizations and the discussion thread in order to record the ontology evolution.

1 Introduction

In this paper we present an innovative approach to develop a domain ontology in collaboration. This approach is synthesized in a groupware application called Co-Protégé.

In this approach, we understand the building of an ontology as a collaborative activity through which a group of experts develops a common knowledge about a domain of interest and where a conflictive situation can arise when different peers have different points of view of the domain.

The management of conflicts or divergencies is useful because conflict negotiation will end up in the improvement of the ontology. It is well known in the field of CSCW that conflict negotiation generates more collaborative interactions among peers.

Consequently, we have conceived Co-Protégé [4] (an extension of Protégé [7]), as a groupware tool which not only allows developing an ontology collaboratively, but also makes the occurrence of a conflict and the process for its resolution explicit. Co-Protégé enables the coexistence of divergent conceptualizations and to follow the discussion thread in order to record the ontology evolution. Co-Protégé integrates the process by means of which the ontology was proposed, discussed, augmented and eventually agreed.

This approach is in the address of other tools that allow the collaborative design of an ontology ([2], [11]) like WebOnto [5] and Apecks [14]. However, this takes into account the asynchronical development of the discussion and of the ontology, and is more focused in the development of a shared ontology than in the development of personal ones as in Apecks. Besides, in most of these systems, the occurrence of divergences is avoided and the management of negotiation mostly kept out of the shared ontology as in WebOnto. Consequently, the shared ontology only considers the last update as in Ontolingua [6]. On the other hand, our focus in the support of the distributed work of the team of the ontology developer, becoming less pretentious than the OntoEdit [12] approach, which proposes a methodology for ontology development.

There are some requirements to take this work to action. The first one is that *the process of the collaborative building of the ontology should be explicit and controlled*. This is to organize and control user actions and interactions, but also to make possible the explicit representation of the conflict solving steps in the knowledge base. A second important requirement is that the *participants in the sharing process should be aware of the shared ontology status and its evolution as well as of the activity itself*. Finally, a last requirement is the need of having a *unified formal model* which allows representing both domain artifacts and the conflicts with their solving steps.

Based on these requirements, we have designed *a process driven groupware system for supporting the collaborative ontology building*.

This paper is organized as follows: first we introduce the process, then we discuss the fundamentals of the Co-Protégé application and the implementation features in section 3 and 4 respectively. Finally, we present some conclusions.

2. The Ontology Building Process

In this section we will introduce the process which is carried out by a group of participants in order to build an ontology collaboratively. This process is an adaptation

Proceedings of the 17th International Conference on Database and Expert Systems Applications (DEXA'06)
0-7695-2641-1/06 $20.00 © 2006 IEEE

of the ks-process (knowledge sharing process) [3] which takes into account the knowledge sharing activity that occurs when a group of participants develops a knowledge repository in collaboration. First we will introduce ks-process and then, we will show how it can be adapted to the collaborative ontology building.

The ks-process. The knowledge sharing activity can be seen as a spiral process where knowledge keeps emerging in each cycle. This activity describes an augmentative building of the common understanding through the contribution of knowledge; even this contribution states a divergent position.

When the knowledge sharing activity is computer-supported by the collaboratively development of a knowledge repository, it is possible to remark that knowledge moves from *private knowledge contexts* to the *community one* and comes back to individuals again. At the same time, knowledge is no longer tacit to become explicit and then it becomes tacit again [9].

In order to capture this, we suggest the ks-process as the one to describe how a group shares knowledge at the same time that it develops its own knowledge repository. This is a spiral process, which is made up of 4 steps: externalization, publication, internalization and reaction.

Externalization is an individual and private activity through which a knowledge unit, which is tacit in the individual knowledge context, becomes explicit as a knowledge artifact in the private knowledge repository. A *knowledge artifact* is the minimal unit of "explicit" and exchangeable knowledge. Some knowledge representation system is needed to make the knowledge explicit which allows modeling k-artifacts and arrange them in the knowledge repository.

Publication is the act of making public some externalized knowledge. It corresponds to the submission of a knowledge artifact from the private to the shared knowledge repository. The act of publishing is generally understood as a *contribution* of a knowledge artifact.

Internalization is an individual and private process, which takes place when individuals realize and acquire the subject of a new contribution. Internalization makes knowledge goes from community knowledge context to the individual one.

Lastly, *reaction* is the act of giving some kind of response to a previous contribution and provoking an "*augmented*" version of the original knowledge. It always involves an externalization and an eventual publication. Reactions can be private, this means that it only produces some change in the individual knowledge context; or it can be public when it is published. Public reactions involve contributions and we call them *contribution by reaction*. Depending on whether the reaction is private or public; there will be a private or public divergence.

There are many causes for a reaction occurrence. It can be either motivated to complement a previous contribution or to give a divergent point of view or just to provide arguments for the original contribution. Thus, reactions enable the occurrence of divergences.

Any reaction is triggered from a previous contribution and all contributions which were triggered by an initial contribution describe a *discussion thread*.

Adapting ks-process to the collaborative ontologies building. For adapting the ks-process to the collaborative ontology building, we have used ontologies to describe the knowledge artifact. Thus, the knowledge-sharing activity ends up in the collaborative development of a shared ontology.

As the externalization step absorbs the impact of using ontologies, externalizing knowledge involves building a conceptualization of a knowledge artifact; now, the *ontological artifact*.

On the other hand, publication means contributing to the shared ontology with an ontological artifact. This contribution involves "integrating" to the shared ontology an ontological artifact. This means that any contribution should be an augmentative version of the shared ontology without introducing any description mismatches [8]. Only "augmentative" contributions should be allowed. But, in case the contribution is divergent (it gives an alternative conceptualization), it is necessary to give account with a mechanism that allows this kind of contribution in order to keep the idea that people can always make a contribution, even if it was divergent.

3 Co-Protégé Features

Co-Protégé is a process-oriented groupware application based on the process that was described in previous section. It is made up of: the *workspace*, which supports the necessary functionalities for externalizing and publishing; the *divergence management component,* which is in charge of making contribution by reactions (divergence occurrences and discussion threads) explicit and lastly the *awareness component* which facilitates internalization.

3.1 The Workspace

The workspace supports the collaborative development of a domain ontology by means of performing externalization and publication actions.

As externalization is a private activity and publication affects the public context, we have conceived a

workspace made up of a public workspace and many private workspaces. The *public workspace* is a shared workspace that is unique and accessible to everyone and contains the shared ontology, and the edition of this ontology is only achieved by publishing ontological artifacts. On the other hand, the *private workspace* is a non-shared workspace (only accessible by its owner). It hosts a *private ontology* which represents the private view of the shared ontology. This ontology is developed by the direct edition of k-artifacts. The private ontology can differ from the shared one, but they can have overlapped parts.

According to the structure of the workspace, the execution of public actions is perceived by any member, but the execution of private actions is hidden from the other members. The main private actions are those to externalize an ontological artifact, they are the required actions to create and update an ontological artifact (Figure 1, 1). The main public action is the publishing action, which allows one to make an ontological contribution from the private to the shared ontology. Besides, there is another public action, the transferring action, which allows defining a private "view" of the shared ontology.

Augmentative development of the shared ontology
Any contribution involves merging the contributed ontological artifact with the shared ontology, but also it should provoke an augmentative version. A contribution is augmentative if it can be *integrated* [8] to the shared ontology without introducing any mismatch in the shared ontology (Figure 1, 2). Therefore, some mechanisms to check the integration viability are required. Each time a publishing action takes place, it is necessary to "check" whether it involves an augmentative contribution. Contributions that pass this checking can be merged to the shared repository without any inconvenience. On the contrary, non-augmentative contributions should be rejected or should be explicitly contributed as a divergent contribution.

3.2 The Divergence Management Component

The divergence management component is in charge of the occurrence of divergences. A divergence can occur in two senses: it can be a private or a public divergence. Private divergences are those that remain private in the private ontology. It is the simplest one and it is easy to support because it is determined by the workspace shape --both ontologies are in two differentiated workspaces (Figure 1,3). On the other hand, a public divergence is a divergence in the shared ontology. This means having alternative ontological artifacts in the shared ontology, in an augmentative fashion. It may be achieved if the underlined model provides the primitives to express them (Figure 1, 4).

In our approach, a non-augmentative contribution is explicitly published as a divergent contribution. Divergent contributions have a special kind of ontological artifact attached: the discussion artifact, which encapsulates a divergent ontological artifact. Particularly, discussion artifacts are the resource through which divergent versions can *coexist* in the same ontology, because they allow encapsulating the inconsistence.

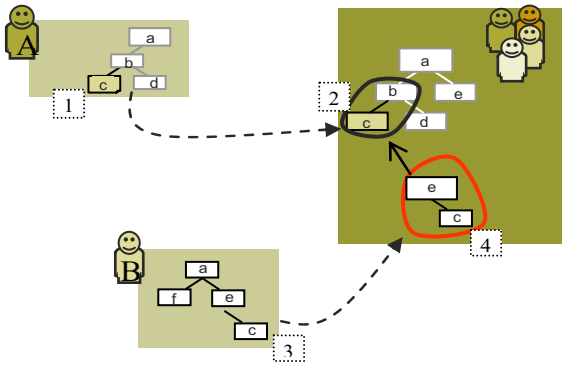

Figure 1. An example

The second goal of the divergence management component is to take divergent contributions to the context of a discussion. The resource to manage this is the *discussion thread*. The discussion thread model provides a simple, yet formal, structure for the discussion and exploration of ontology building. It is a conceptual model that identifies the matrix of logical elements that represent the different kind of contributions by reaction. Discussion thread is in charge of linking the discussion artifacts as well it is in charge of identifying the role of the contributed artifact in the context of the discussion (initial artifact, augmentative artifact, divergent artifact and argumentations). The discussion thread is an aggregation of augmentative and/or divergent contributions. In figure 1, in the shared side, both rounded shapes which are linked by an arrow represent the discussion thread. The discussion thread also holds the argumentations which are attached to the contributions and allow carrying out negotiation of a conflictive contribution by discussing about different positions.

3.3 Capturing the Activity Knowledge

While the collaborative activity is carried out, knowledge about this activity has to be captured in order to maintain the activity history and to improve the collaborative activity. On the top of the workspace, the

activity knowledge is captured in terms of: performed actions, domain knowledge, people and the relationships among them. This knowledge is ontologically represented in generic ontologies (Figure 2).

To capture knowledge about the performed actions it is necessary to track each event that occurs in the workspace. This knowledge will be part of the *action ontology*. The action ontology represents knowledge about the performed action (*action*), who performed this action (*member*) and which *artifact* it involves. It also covers knowledge about the discussion activity.

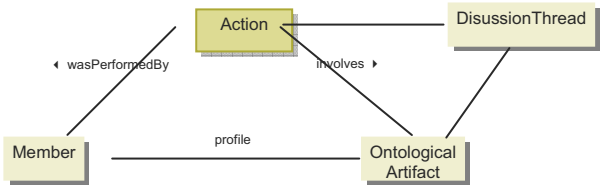

Figure 2. Schematic relationships among the domain and generic ontologies.

In order to complete the knowledge about the activity, the system also captures knowledge about members by means of the members' profile ontology and knowledge about conflict solving process through the discussion thread ontology. In the user's profile, people indicate what actions, ontological artifacts and users and other users' activity they are interested in

3.4 Awareness Services

Awareness is a relevant component of any groupware application; it keeps users up-to-date about the collaborative activity [10] that is, it will keep people aware of the ontology changes and discussion evolution. Awareness service can be seen as facilitator of the internalization (which promotes the reaction) because it is in charge of reporting about new contributions and the identification of highly-active concepts. But awareness is also useful to complement the support of divergences, since it makes the divergence occurrence evident.

As any awareness services, it gathers information about the activity and then delivers it to the user. For tracking the collaborative activity, people need information in the context of history of the activity; this is low and high level information. Low-level information is already captured in the action ontology. High-level information could be deduced by mining the low-level information (Who has been contributing with this person? How has this ontological artifact evolved? What are the more active topics?). High-level information is also useful to update the action ontology by adding high-level actions and also to update the member profile by adding new discovered interest.

Co-protégé awareness mechanism tracks every contribution action. Each time a new action takes place; Co-Protégé captures information about the performed action and stores it in the knowledge base. Then people are notified. The Co-Protégé awareness mechanism delivers this information through *notifications*. The notification mechanism only delivers those notifications that answer the user's profile. A notification is related to the action and attached to the users. If the user is on-line, he/she is immediately notified of the occurrence of a new event, otherwise he/she will be notified the next time he/she logs in.

Awareness services also keep users aware of the changes between the two ontology versions, in order to provide awareness of private divergence.

4. Co-Protégé System

Co- Protégé is a set of plug-ins that extends Protégé. Preserving the Protégé-2000 developing philosophy, Co-Protégé adds functionalities to edit ontologies and knowledge bases in a collaborative fashion. Co-Protégé visualization is in terms of tabs like in Protégé-2000 (Figure 3). There are tabs for modeling the shared-private workspace, the conflict tab, the user tab and the difference tab.

Shared-Private Workspace Tabs. Co-Protégé proposes tabs that "overlap" both workspaces in the same tab in order to make easily achieving to a direct manipulation of the two ontologies. Only the private side (on the left) has the same functionality as the Protégé-2000 to edit the private ontology; the shared side on the right) cancels them (the shared ontology is only updated by publications). There is one tab of this kind for each kind of frame (class, slot and instance). A conflict is created in the shared-private tab by selecting the set of frames that will be put in conflict. After that, the frames are shown "in conflict".

The system makes incompatibility checking each time a publication is performed in order to ensure an augmentative contribution. Whatever the checking result may be, Co-Protégé informs it at the bottom of this tab.

User Tab is to manipulate the user profile. Users' interest can point to any kind of frame described by the metamodel of Co-Protégé, that is, elements of the shared ontology, other users, conflicts and conflict components. There are some cases where the system is able of updating the user's profile or making suggestions [1].

Conflict Tab defines a space where users can browse through the conflict and make it evolve. Once a conflict was created, it becomes part of the conflict list, where all currently open conflicts are enumerated. Users can add alternatives and argumentations. Alternative are created with frames from the private ontology. This is the

mechanism that allows for the publication of divergent contributions.

Co-Protégé supports two visualizations of notifications. One indicates the degree of similarity/difference that ontological artifacts in both ontologies maintain in order to provide awareness of private divergence. This is shown over the private ontological elements. This visualization is rendered each time any change occurs at the private or shared ontology. The other visualization is more general and shows all the notifications in a chronological order. At the user's tab, users can specify different filters to show notifications.

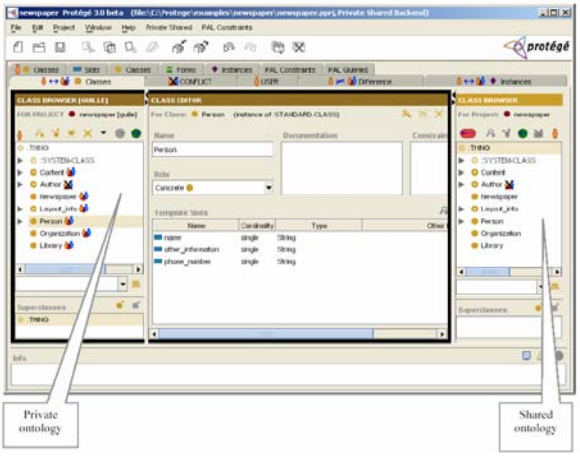

Figure 3. A snapshot of Co-Protégé. Both private and shared ontologies can be appreciated simultaneously.

Co-Protégé Implementation Features. Co-Protégé extends Protégé through the definition of some plugins by following Protégé extension philosophy. In Co-Protégé a project is made up of the shared ontology plus all private ontologies (one for each user). Co-Protégé is a client-server application, where a project is defined as a Protégé's metaproject. In this metaproject every ontology is defined (the shared and each private) together with the access permissions.

Co-Protégé uses the Protégé-knowledge model: classes, slots, facets and instances. However, it extends Protégé-2000 metamodel in order to provide primitives to model the primitive frames of a shared ontology and conflict primitives. Besides, Co-Protégé defines a set of generic ontologies to model the knowledge about the users and the collaborative activity.

5 Conclusion and Future Work

Co-Protégé is groupware tool which not only allows the collaborative development of an ontology, but also makes explicit the occurrence of conflicts and the conflict resolution process. Co-Protégé stores in the knowledge base knowledge about the ontology building activity in order to help people to understand the evolution of the ontology. Although this approach seems very promising, it still needs to be evaluated. In this way, our immediate objective is to get more feedback about the use of the tool by real users. To do so, we are actually working on an enhanced version of the prototype that will support OWL. We plan to release our tool as an open-source plugin for Protégé in a near future.

6. References

[1] Baldo G., Lemus M., Diaz A. Acciones como Fuente del Conocimiento de una Comunidad. Clei'03, La Paz, Bolivia, Octubre. 2003.

[2] Corcho O., Fernández-López M., Gómez-Pérez A. Methodologies, tools and languages for building ontologies: Where is their meeting point?. Data Knowledge Eng. 46(1): 41-64. 2003.

[3] Diaz, A., Canals G. Divergence Occurrences in Knowledge Sharing Communities. LNCS, Springer Verlag –Proc.of Criwg'04. 17-24. 2004.

[4] Diaz, A., Baldo G. CO-Protégé: A Groupware Tool for Supporting Collaborative Ontology Design with Divergence. Proc. of the 8th International Protégé Conference, July, 18-20th, Madrid, Spain. 2005.

[5] Domingue J., Tadzebao and Webonto: Discussing, Browsing and Editing Ontologies on the Web, in: Proc. 11th Knowledge Acquisition Workshop (KAW98), 1998.

[6] Farquhar A., R. Fikes, and J. Rice; The Ontolingua Server: a Tool for Collaborative Ontology Construction; Intl. Journal of Human-Computer Studies 46, 1997

[7] Gennari J., M.A. Musen, R. Fergerson. The Evolution of Protégé: An environment for knowledge-based systems development. IJHCS, vol 58 (1): 89-123. 2003.

[8] Klein, M. Combining and relating ontologies: an analysis of problems and solutions. In Gomez-Perez, A., Gruninger, M., Stuckenschmidt, H., and Uschold, M., eds, Workshop on Ontologies and Information Sharing, IJCAI'01, Seattle, USA. 2001

[9] Nonaka, I. and Takeuchi, H. The Knowledge-Creating Company, Oxford University Press. 1995

[10] Schmidt K. The Problem with Awareness Introductory Remarks on Awareness in CSCW Journal 11(3-4), 285_298. Kluwer Academic Publishers. 2002

[11] Staab S., R. Studer (eds.). 2004. Handbook on Ontologies. International Handbooks on Information Systems, Springer Verlag,

[12] Sure Y., M. Erdmann, J. Angele, S. Staab, R. Studer, D. Wenke, OntoEdit: collaborative ontology engineering for the semantic web, in: First International

[13] Semantic Web Conference (ISWC02), LCNS vol. 2342, Springer, Berlin, 2002, pp. 221-235.

[14] Tennison J., K. O'Hara, N. Shadbolt: APECKS: using and evaluating a tool for ontology construction with internal and external KA support. IJHCS. 56(4): 375-422. 200