Ontology mapping in community support systems

Martin S. Lacher, Wolfgang Wörndl, Michael Koch, Harald Brede
Institut für Informatik, Technische Universität München, Germany
{lacher|woerndl|kochm|brede}@in.tum.de

May 12, 2000

Abstract

Communities have been shown to be a very valuable source of knowledge. Thus, community support has become an important issue for Knowledge Management efforts. Community support systems must reflect the loose coupling of community members as well as the heterogeneity of communities. The heterogeneity of community members brings with it a semantic heterogeneity of terminology and perspective on the available knowledge in the community. A solution for this problem lies in the use of a centralized community ontology on which community support services can be based for semantic interoperability.

We introduce an agent-based architecture for community Knowledge Management support. Exemplary support services are presented and it is shown how those services can be implemented, based on a centralized ontology. From a usability perspective, however, an approach with a centralized ontology has a number of disadvantages. We will discuss these disadvantages and present a concept for community support with distributed ontologies. The distributed ontologies paradigm is shown to be superior in terms of usability for several theoretical and practical reasons. To conserve the semantic interoperability achieved with a centralized ontology, we present a technique for automated mapping between distributed ontologies. The mapping technique is based on a method for automated web page categorization.

1 Introduction

Information overload and the problem to find task-relevant information are topics raised a lot in recent publications and can be experienced in everyday work with computer based information systems (e.g. the Internet). As a consequence to overload with irrelevant information and lack of relevant knowledge, Knowledge Management emerged as an area for research. The holistic approach of Knowledge Management includes a psychological, a business and an IT perspective. IT supports Knowledge Management efforts in that it supplies the basis for knowledge communication.

The most valuable resource for relevant information is the knowledge of experts in a field or the information collected by people interested in the same field. Therefore, these communities have become important for Knowledge Management, and community support tools that support the collaborative collection, storage and re-use of information have gained a lot of attention.

For community support tools, a much higher interoperability among different applications is required than for groupware systems. However, the current systems still lack the possibility to exchange data with each other. A user has to log on to different online communities manually and provide his profile information again and again, and there is no possibility to distribute new information to different communities in one step.[1]

There is a tremendous need for standards, a common scalable infrastructure as well as common categorization schemes to manage community information. In this paper we will present our ideas for a community support infrastructure and focus on the role ontologies play in this infrastructure and how we envision the implementation of these ideas.

The remainder of this paper is structured as follows. Section 2 provides the context of community support in Knowledge Management. Section 3 describes our community support architecture and gives an overview over its ontology-based services. In Section 4, a definition of an ontology is given. The centralized and the distributed approach to ontology support are discussed and a concept for distributed ontology support in our framework is presented. Section 5 describes an ontology mapping technique as well as some example services for Knowledge Management that take advantage of the mapping. The results of the paper are summarized in the conclusion in Section 6.

2 Community support and Knowledge Management

In general a *community* is a group of people who share some interest or belong to the same context. So a community can be seen as a describing identity for a set of people. Mynatt et al. concretize a bit more:

"[A community] is a social grouping which exhibit in varying degrees: shared spatial relations, social conventions, a sense of membership and boundaries, and an ongoing rhythm of social interaction" [1].

Examples for communities are all students in a university department, all inhabitants in a neighborhood or all people interested in collaborative filtering.

As mentioned in the introduction, (virtual) communities are a good source for relevant knowledge. To foster the flow of knowledge [2], community support systems that support the exchange of tacit and explicit knowledge among community members are

[1] The exchange problem and the problem of reusing user profiles is not a unique problem to community support systems, but a general problem in disconnected information systems.

required. Further, social structures and customs in communities can be exploited to provide and use valuable knowledge about the community itself.

The basic support concepts for communities are:

- Provide a medium for direct communication and for exchange of comments on objects within the common scope of the community. This information about the structure of the community may be collected and stored for later reference and retrieval.
- Discover and visualize relationships (membership in the same community, existence of common interests). This can support people in finding possible cooperation partners for direct interaction (*matchmaking, expert finding*).
- Use the knowledge about relationships to perform (semi-)automatic filtering and personalization. This helps to reduce the search effort and enables to deal with the information overload.
- Allow the collaborative creation of common schemes for information classification (*ontologies*) that can support retrieval of relevant information.

There are already several systems that are implementing aspects of these basic support concepts. News- and Chat-systems (including different kinds of community networks) provide a place to meet and a communication medium. Buddy systems like ICQ or the AOL Instant Messenger provide detailed awareness information [3]. Online communities provide a place to communicate, awareness and a rich functionality for storing and retrieving (community) information. Recommender systems like Movie-Critic[2], Knowledge Pump [4, 5] or Jester [6] do matchmaking on the basis of user profiles and then provide recommendations based on ratings of other community members. Other systems like Referral Web [7] and Yenta [8, 9] focus on expert finding and explicit matchmaking.

The problem with all these systems is, that they do not offer standard interfaces and are not connected. Usually, a person is member of several communities. People have to enter their profile information and community information again and again and have to interact with different applications.

Thus, better interoperability among community support applications like recommender systems and online communities is needed. Our approach to this problem is to provide a general infrastructure and reusable components for community support systems.

3 The community support architecture

A community support infrastructure has to support communication among different communities and between users and communities. Support has to enable users to contribute both profile information and content to communities easily.

[2] See http://www.moviecritic.com/

In this section we propose an agent based infrastructure with agents that are organized in user and community agencies.[3]

Information and services that are private to the user are collected in the *user agencies* which are located near the user, and team and community relevant information and services are collected in the *community agencies*. Figure 1 shows this general idea.

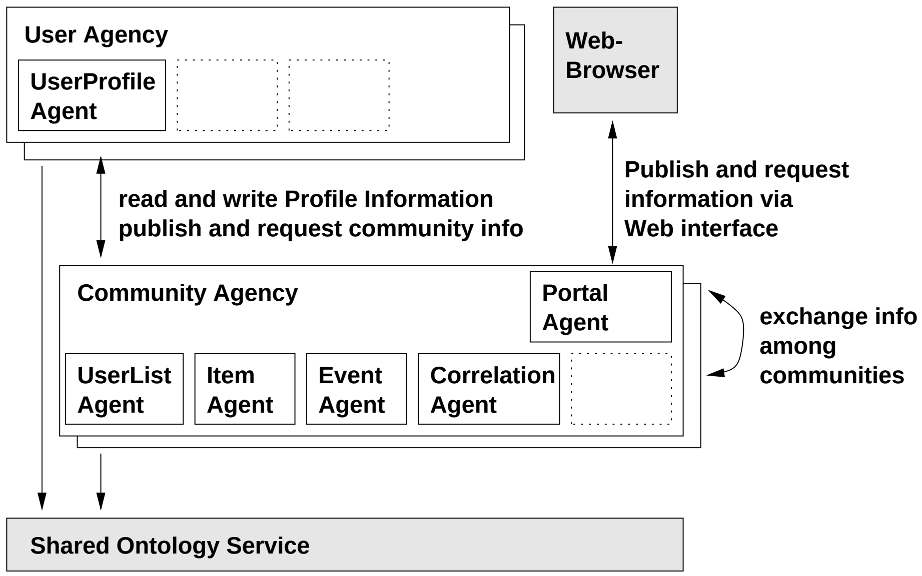

Figure 1: Community Support Architecture

In the project Cobricks[4] we are currently exploring these ideas.

3.1 Agents

Basis of our infrastructure is a software agent platform. The agent paradigm provides for a good modularization scheme of services and loose coupling among functional components (agents). This assures an open architecture and easy integration of future components.

Communication among agents is done using FIPA ACL, the agent communication language defined by FIPA[5]. An agent communication language defines the basic structure of messages between agents with precisely defined syntax, semantics and

[3] Communities are represented by community agencies, users are either represented by user agents or they directly access community agencies via Web interfaces. Community support tools are collections of APIs or user interfaces provided by a community agency.

[4] See http://www11.in.tum.de/proj/cobricks/

[5] Foundation for Intelligent Physical Agents; for more information see http://www.fipa.org/

pragmatics[6]. This makes it possible to use the language to communicate between independently designed software agents.

Hence, the basic infrastructure primarily takes care of modularization and communication among the modules (agents). Additional parts of the infrastructure are a standard directory service (LDAP/X.500) for finding agents and a set of security components that take care of authorization and access right checking.

For the content of messages a shared ontology [10] is used as the basis for a common semantic understanding of exchanged information.

3.2 Ontology

An ontology in our architecture is understood as a directed graph concept graph, within which the nodes represent subject matter classes identified by unique natural language terms. Through the ontology, a set of concepts in a knowledge domain is characterized and the corresponding taxonomy is specified. A normed taxonomy for a community assures that content that is communicated through agents is understood everywhere semantically correct. On the other hand, a community ontology also assures that community members have a common semantic basis for communication without misunderstandings. For the description of our architecture, we will assume one centralized community ontology, which can be understood as a directed graph (Figure 2) for information classification (e.g. like the directory structure of YAHOO^{TM}). The ontology concept in our architecture is explained in more detail in Section 4.

3.3 User Agency

The user agency provides means for managing the user profile and for requesting and posting information from and to community agencies. The central agent here is the *profile agent*.

The profile agent keeps a qualification/interest profile of the user. The profile stores demographic information, relationship information, ratings, and links to information contributed to communities. For semantic interoperability of profile parts such as qualification and interest profile, we rely on the shared community ontology. We implement simple user interest and qualification profiles by classifying the user into one or more classes in the ontology concept graph. With each classification of the user, an expertise level attribute is attached (Figure 2). Classification of a user in a high-level concept represents overview knowledge, whereas classification in a more specific concept represents detail knowledge. The expertise level attribute denotes how well the respective user knows a concept and related issues. For example, in Figure 2, user Smith has an excellent overview of computer science and is an intermediate when it comes to intelligent agent technology.

There are several other agents in the user agency, which assist the user in requesting information from and providing information to the community. Information provided

[6] For the semantics and pragmatics part agent communication languages usually make use of speech act concepts.

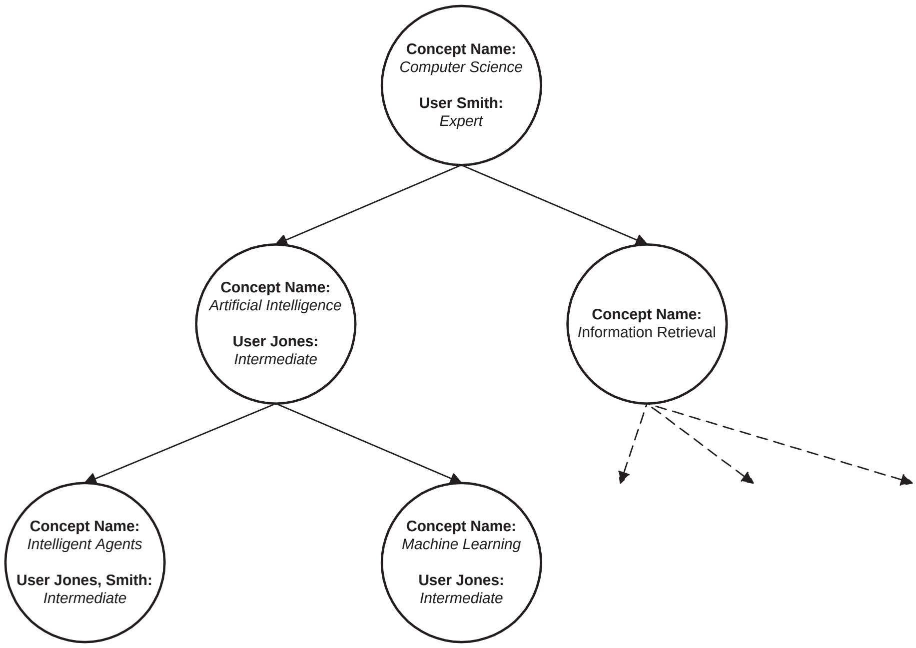

Figure 2: Community ontology with user profile attributes

to the community is classified according to the community ontology for later retrieval. Information available in the community has been classified before and thus content-based filtering based on the semantic meta-information in documents can be performed. The user is provided with documents which are relevant to her current work context or comply with her qualification. Personalized information can be pushed to or pulled by the user (see [11] and [12] for more details on this).

3.4 Community Agency

The main task of a community agency is to support information exchange among users. It provides functions to publish information and comments, to request published information and comments either through a query (pull) or through a subscription (push). Information handled by the community agency can be explicit knowledge of interest for the community, meta information on information items and structural information on the community (e.g. on users).

Hence, one part of a community agency are repositories where information contributed by the members is stored. This can be basic information or references and annotations. All the information and annotation handling as well as categorization is provided by the item agent. The community agency makes its information available to other agents through the agent infrastructure. Additionally, we have implemented community portal agents that provide web portals for users to connect and authenticate via a web browser.

Other agents in the community agency provide recommendations or perform collaborative filtering. In our agency we have implemented one agent that calculates user correlations and another one that makes use of these correlations to calculate recommendations. Correlation calculations are also the basis for an expertise brokering agent. A query for a certain qualification profile based on the community ontology is performed and if found, a respective user identity is returned.

The workspace awareness service provides information on newly created knowledge in the community. Both, the user profile and the knowledge available in repositories in the community is based on the ontology. By matching community repository classes with interest profiles and pushing information about changes in the repository to the user, she can be kept up-to-date with the creation of knowledge in the community.

To provide structured access to the community information, browsing and searching services are based on the community ontology.

For more detailed information on the Cobricks framework, see [11] and [12]. Large parts of the architecture and the defined services have been implemented and are undergoing testing at the moment.[7]

One of the central points that the work with Cobricks showed, is that ontology support for communities is best realized with distributed ontologies instead of one centralized ontology. We will give reasons for this and present a concept for distributed ontologies in the next section.

4 Distributed ontologies for community support

The usage of ontologies to support semantic interoperability in KM has been widely accepted in research [13]. However, only very little research has been published on how to put ontologies into productive use for communities [14]. Communities have very distinct organizational structures and community support IT systems have to reflect this. This also applies for services based on ontologies. Existing research considers ontologies as an objective conceptualization of a real world domain. The possibility of subjective conceptualizations has never been discussed.

In the previous section, a number of applications based on a community ontology have been presented. In this section, we will explain in more detail, what the definition of an ontology is in our community support architecture. We will further motivate why several distributed ontologies are necessary for community support and what they represent. Finally we will introduce our concept for distributed ontologies in a community support system.

[7] See http://www11.in.tum.de/proj/cobricks/

4.1 Definition of an ontology

Formally, an ontology is an explicit representation of a conceptualization of a knowledge domain. More informal, an ontology can be visualized as a concept graph, within which the nodes represent real world concepts and the edges represent relations between the concepts.

A taxonomy is usually considered as the set of natural language labels for the concepts of an ontology. Thus, in the concept graph, the taxonomy would be the set of all labels assigned to the nodes. If an ontology is reduced to a taxonomy, a common semantic understanding of a concept designated by a certain label is assumed and the concept is not explicitly specified.

For our community support architecture, we narrow the notion of ontology introduced in [15] down to a simpler taxonomy structure, as shown in figure 2. The main goal of ontology design here is the definition of a common taxonomy for the annotation of community items and profile information. More complex ontology definitions allow more elaborated support applications, such as automated resoning [14, 16, 17, 18]. Formal aspects of ontologies have been extensively discussed in [10, 19].

Our simple ontologies can be represented by directed concept graphs within which the nodes represent subject matter classes identified by unique natural language terms (e.g. *AI, Intelligent Agents*). The edges represent subsumption relations (e.g. *Intelligent Agents subclass_of AI*). The nodes can have multiple parents to allow shared concept classes (e.g. *Machine Learning* could be a child of *AI* as well as *Information Retrieval*).

An information item can be classified into several classes (e.g. a book that is about *AI* and *Information Retrieval*). However, it is not allowed that an item is classified into several classes on the same path from the root to the leaves of the graph. Annotations about the qualification of users for the user profile as described in Section 3 can be made in any node of the graph.

4.2 The need for distributed ontologies

Cobricks was first designed with one central community ontology. There are several reasons why this proved impractical and a concept with several distributed ontologies is preferable.

An ontology is essentially a perspective on a real world domain. The same concept in the real world can be seen from different points of view by different people. The perspectives are different in the set of predicates that is assigned to a concept to describe it. This in turn has a change in the connections between the concepts as a consequence. Another obvious difference in perspectives shows in different taxonomies used for the same concepts (Figure 3).

Thus, a centralized ontology represents a global objective perspective on a domain. All users in the community have to have the same perspective on the information managed in the community. Systems following the centralized ontology paradigm (the *god's eye paradigm* [20]) make a number of (not always explicitly stated) assumptions. Most of these assumptions prove as disadvantages as we will explain here.

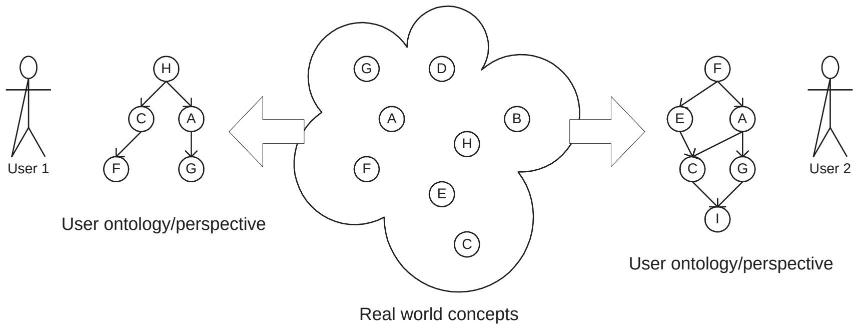

Figure 3: Ontologies as perspectives on a domain

The centralized approach neglects the fact that people can have very different subjective perspectives on domains of knowledge. These subjective perspectives are all valid points of view and cannot be dubbed *wrong* with respect to the central ontology. Subjective perspectives are not totally neglected in a centralized approach (Figure 4). They are much more assumed to be integrable into the central ontology and represented by connections in-between concepts. Undoubtedly, the construction of an ontology which integrates all possible existing perspectives is a hard task [21]. Moreover, future or evolving perspectives cannot be integrated at all. With a centralized ontology, a user's personal perspective on a domain is not stored and may not be made aware to other community members. However, these subjective user perspectives are often very useful and contribute to new insights.

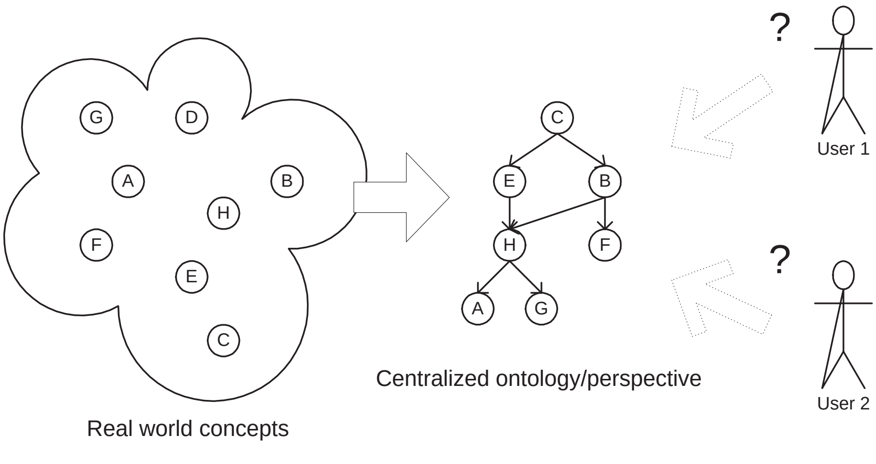

Figure 4: A centralized ontology - one centralized perspective

Looking at the Cobricks architecture, practice showed that it is convenient for users to use their own personal ontology as a perspective on the domain of interest. The reason for this is a higher trust in the ontology structure and a better usability [22]. With a centralized ontology, the user must learn and understand the centralized ontology scheme. She may not agree, however accepting the centralized ontology is the only means to exchange knowledge with other community members. Concept queries based on the ontology would largely depend on how well the user posing the query as well as the user who classified information understands the ontology. Chances are, that their understanding differs.

Depending on whether a centralized ontology is created in a central top-down manner, or in a distributed bottom-up manner, the user may show varying levels of distrust in the structure of the ontology. A centrally created ontology would seem unnatural and forced. The trust of users in a centralized scheme would be minimal and so would participation be. Moreover, it is questionable, if a central authority would have the expertise to create a centralized objective ontology.

4.3 A concept for distributed ontologies for community support

The notion of distributed ontologies has first been introduced in [20]. We show that several distributed ontologies are necessary for community support. In order to allow each user to organize her information, each user requires a user ontology. Moreover, for each community to allow for exchange of information, a community ontology is necessary. And for all information that is not directly related to communities, as well as a basis for communication among communities, a global ontology is required. Each of the ontologies also represents a perspective on other ontologies.

This approach with several distributed ontologies has several advantages. First of all, the trust of users in a personal, self-constructed ontology will be much higher than with a central ontology. This is an essential point for KM support tools [22].

The user can organize information without the overhead of understanding the centralized ontology. With the classification of information according to the personal ontology, the user's perspective on the information is conserved as well. This perspective can be of value for the whole community.

The retrieval precision of stored information can be increased, since the user is exactly familiar with the concepts in her personal ontology. If a query is performed for a concept which is not in the user's ontology, there is no disadvantage, since the concept must be learned in any case.

For the user, information exchange is much easier than with a central ontology. If the user provides information for the community or others, she just has to classify the information according to her personal ontology and make it publicly available.

The community ontology has to be constructed in a bottom-up manner by community members to increase trust in the ontology [8]. The construction may be moderated

[8] This is for example successfully applied in the Intraspect Organizational Memory (See http://www.intraspect.com/)

for larger communities and should start with a basic skeleton ontology that can be based on the Open Directory Project [9].

The global ontology does not necessarily have to be created in a bottom-up manner. However it is an advantage, since only the exponentially growing number of Internet users can scale with the exponentially growing number of documents on the Web.

The concepts for distributed ontologies here rely on a mapping that is performed between the user ontology concepts, the community ontology concepts and the global ontology concepts. We will explain how this mapping is performed in the next section.

5 Community ontology mapping

In the previous section we motivated why distributed ontologies are necessary for community support. We showed that a mapping between ontologies is required. In this section we will motivate the mapping further. We will also present a mapping between ontologies based on a web page categorization technique introduced in [23].

Working with personal ontologies is convenient for the user. If the user finds a new document of interest for her, she classifies it according to her personal ontology for later retrieval. However, the user is also a member of a community, the members of which have shared interests with the user. Naturally, the documents classified by the user would be of interest for other community members and vice versa. Thus we have to provide a mapping between the concepts of the different distributed ontologies. The mapping can be performed manually by classification of a document with respect to all distributed ontologies. Corresponding classes are the classes in different ontologies to which the document has been assigned. However, a manual mapping would jeopardize the advantages of distributed ontologies. Moreover, the mapping would most likely be inconsistent. Different users will assign the same document to different classes and the classification by a user itself may be inconsistent and guided by mood. The concept we introduce here performs automatic mapping between ontologies.

5.1 Probabilistic mapping calculation

A probabilistic mapping between ontologies can be calculated if the same documents are classified in several ontologies. Let $D = d_1, d_2, ..., d_n$ be the set of documents in class c of a user ontology. C describes the event that $d_i \epsilon c$. d and e are classes in a global ontology and D and E are events analog to C. If m of the n documents in class c are in class d of a global ontology and $l = n - m$ documents in class e of a global ontology, then $P(D|C)$ can be estimated as $P(D|C) = \frac{m}{n}$ as well as $P(E|C) = \frac{l}{n}$.

Mapping between concepts in two ontologies is easy, if the same documents are already classified in both ontologies. This can be the case for the mapping from a user ontology to a global ontology. It is likely that documents classified by the user with respect to her ontology have already been classified with respect to the global ontology.

[9] See http://www.dmoz.org/

If the same documents are not classified in two ontologies already, we take the set of documents from a class in the user ontology and assign each document to a class in another ontology (community, global, other user) by automatic text categorization. The only assumption that is made, is that the target ontology and especially natural language identifiers for the concepts have to exist.

5.2 Web page categorization by context

Our text categorization is based on a concept presented in [23], where web documents are classified by context instead of content. For categorization of a page p, information from pages referring to p by hyperlinks is used. This method of automatic categorization is based on the following assumptions:

- a web page which refers to a page p contains information about p's contents
- the surrounding text of a link to p supplies descriptive information on p
- those hints are sufficient to classify p

This classification by context instead of content simplifies the classification task. Firstly, the context of a link is easier to extract than the relevant content of a web page. Moreover, the link context is usually a short and precise description of the contents of the destination page. This is an elegant way around the problem of the reduction of the feature space with many common text classification techniques [24].

The information used for categorization consists of the URL, the text for this link (anchor), the text surrounding the link, the title of the page on which the link is found, etc. . All of this information is tagged by HTML tags and can thus be easily extracted from the page. The extracted information is grouped into *URL Context Paths* of the form (URL:C1:C2:...:Cn), where C1 is the text of the anchor of the URL, C2..Cn are the enclosing contexts (e.g surrounding text or the page title) in nesting order. From the context elements, noun phrases are extracted from the contexts using stem reduction. For example from the phrase ”The world wide web has become an impressive open structure for sharing of information” the noun phrases ”world wide web”, ”impressive open structure” and ”information” would be extracted. The LTPOS tagger used for this [10]. The noun phrases are weighted according to their distance from the link to the destination page.

These context paths are matched against ontology concepts with identifying concept names. For the concept names, a weighted word neighborhood is created using a thesaurus to allow for slight deviations in the taxonomy. The results of the matchings are conditional probabilities for class mappings.

The problem with this technique is, that a single page cannot be categorized, but instead a number of referring pages is required. We have two solutions for this problem. First, for each new page that is classified in one of the community ontologies, the context path is stored in a database. If a new page has to be categorized, the context

[10] See http://www.ltg.ed.ac.uk/software/pos/

paths in the database are searched and if context paths for this URL can be found, the page can be classified. If no stored context paths are found, the web is searched for pages referring to the destination page. This can be done most effectively using a meta-crawler. A number of pages will have to be analyzed for context information in this case, since a lot of pages have links without any description.

If no matching page, containing sufficient information is found, the context path has to be simulated. As described above, each user manually classifies pages according to his personal ontology. Instead of creating the context path through link-contexts, the path through the user ontology can be used, since the ontologies are directed graphs. The first sentences of the document, including title, are used for additional classification information and reduced like normal contexts. The usage of a word neighborhood table keeps the error rate low.

The conditional probabilities required for the mapping can now be calculated the same way as shown above. Depending on the purpose that the ontology mapping is used for later, these conditional probabilities can now be utilized.

5.3 Utilization examples for an ontology mapping

The conditional probabilities for the ontology mapping can now be utilized for important processes for KM in communities.

One important process is the retrieval of explicit knowledge in the form of documents. We assume as an example that a user is interested in Web Mining and has classified a number of documents on that topic in a class in his ontology. The user would like to retrieve new documents on this topic now and poses a query on the concept "Web Mining" to the community ontology. The ontology mapping resulted in $P(\text{"Text Mining"}|\text{"Web Mining"}) = 0.8$ and $P(\text{"Hypermedia"}|\text{"Web Mining"}) = 0.2$, meaning that 80% of the documents classified by the user as "Web Mining" are classified as "Text Mining"in the community ontology and the rest as "Hypermedia". In this case (for document retrieval), the probabilities would be ignored and new documents from both community ontology classes would be returned. It is thus assumed that the user would not want to miss information classified by the community. If the same query is posed to the global ontology, the user could manually override this default and only retrieve information from the class with the highest probability.

Another important process for KM in communities is the publication of knowledge by users to the community. In this case, to ensure a consistent classification, only one destination class must be chosen. The published document would be classified into the community class with the highest conditional probability.

A key issue for KM in communities is the provision of awareness about the structure of the community. If a user would like to know what the community considers related to the concept "Web Mining" in his personal ontology, she can browse the community ontology around the concepts "Text Mining" and "Hypermedia", which are the mapping results. Again, in this case, all mapped classes are of interest.

Direct communication of knowledge between users is another important process to be supported. This can for example be performed by agents on behalf of the user. To prevent semantic misunderstandings during the exchange of documents, an ontology mapping between the two user ontologies is performed. If the set of common classified documents is too sparse, the mapping can be performed through the detour of mapping to the community ontology. In this case, to ensure consistency, only the class with the highest mapping probability must be used.

6 Conclusion

We briefly introduced an architecture for community Knowledge Management support. The services offered by the agent-based architecture have been presented and it has been shown how the implementation of most services is based on an ontology. We showed that, in terms of usability, an approach with distributed ontologies is more suitable for community support than an approach with a centralized ontology. With a distributed approach, however, a mapping between ontologies is required to conserve semantic interoperability. We introduced a solution for automated mapping between ontologies based on a web page classification technique. The chosen classification technique classifies web pages by context (referring pages). We have given examples for the utilization of the ontology mapping in important processes for community Knowledge Management support such as knowledge communication and provision of community context awareness. The presented mapping technique relies on explicit relations between artifacts (e.g. links between web pages) classified with respect to the ontologies. An issue for future research remains how a mapping can be performed in case there are no related artifacts.

References

[1] E. D. Mynatt, A. Adler, M. Ito, and V. L. ODay, "Design for network communities," in *Proc. ACM SIGCHI Conf. on Human Factors in Compt. Syst.*, 1997.

[2] U. M. Borghoff and R. Pareschi, Eds., *Information Technology for Knowledge Management*, Springer Verlag, Berlin, 1998.

[3] J. Michalski, "Buddy lists," *Release 1.0*, , no. 6, Jul. 1997.

[4] N. Glance, D. Arregui, and M. Dardenne, *Knowledge Pump: Supporting the Flow and Use of Knowledge in Networked Organizations*, Springer Verlag, Berlin, 1998.

[5] N. S. Glance, D. Arregui, and M. Dardenne, "Making recommender systems work for organizations," in *Proc. Intl. Conf. on Practical Application of Intelligent Agents and Multi-Agents (PAAM)*, Apr. 1999.

[6] K. Goldberg, D. Gupta, M. Digiovanni, and H. Narita, "Jester 2.0: Evaluation of a new linear time collaborative filtering algorithm," in *Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, Aug. 1999.

[7] H. Kautz, B. Selman, and M. Shah, "Referral web: combining social networks and collaborative filtering," *Communications of the ACM*, vol. 40, no. 3, pp. pp. 63–65, Mar. 1997.

[8] L. N. Foner, "Yenta: A multi-agent, referral-based matchmaking system," in *First International Conference on Autonomous Agents*, Feb. 1997.

[9] L. N. Foner, *Political Artifacts and Personal Privacy: The Yenta Multi-Agent Distributed Matchmaking System*, Ph.D. thesis, Massachusets Institute of Technology, Jun. 1999.

[10] T. R. Gruber, "Ontolingua: A mechanism to support portable ontologies," Tech. Rep., Stanford University, 1992.

[11] M. S. Lacher and M. Koch, "An agent-based knowledge management framework," in *Proc. AAAI Spring Symposium 2000*, Mar. 2000.

[12] M. Koch and M. S. Lacher, "Integrating community services - a common infrastructure proposal," in *Proc. Knowledge-Based Intelligent Engineering Systems and Allied Technologies*, Aug. 2000.

[13] V. R. Benjamins, D. Fensel, and A. G. Pérez, "Knowledge management through ontologies," in *2nd Int. Conf. on Practical Aspects of Knowledge Management (PAKM98)*, U. Reimer, Ed., October 1998.

[14] S. Staab, J. Angele, S. Decker, M. Erdmann, A. Hotho, A. Mädche, H. Schnurr, R. Studer, and Y. Sure, "Ai for the web - ontology-based community web portals," in *AAAI2000/IAAI2000 - Proc. 17th National Conf. on Artificial Intelligence and 12th Innovative Applications of AI Conf.* Jul. 2000, AAAI Press/MIT Press.

[15] FIPA, "Fipa 99 specification," Tech. Rep., FIPA, 1999.

[16] D. Fensel, J. Angele, S. Decker, M. Erdmann, H. Schnurr, S. Staab, R. Studer, and A. Witt, "On2broker: Semantic-based access to information sources at the www," in *World Conference on the WWW and Internet (WebNet 99)*, October 1999.

[17] D. Fensel, A. Witt, J. Angele, S. Decker, M. Erdmann, H. Schnurr, S. Staab, and R. Studer, "On2broker in a nutshell," in *8th World Wide Web Conference*, May 1999.

[18] S. Staab, J. Angele, S. Decker, M. Erdmann, A. Hotho, A. Mädche, H. Schnurr, R. Studer, and Y. Sure, "Semantic community web portals," *Computer Networks*, May. 2000, Special Issue: WWW9 - Proc. of the 9th Intl. WWW Conf, Amsterdam, The Netherlands.

[19] T. R. Gruber, "Toward principles for the design of ontologies used for knowledge sharing," Tech. Rep., Stanford University, August 1993.

[20] M. Bonifacio, P. Bouquet, and A. Manzardo, “A distributed intelligence paradigm for knowledge management,” in *2000 AAAI Spring Symposium, Bringing Knowledge to Business Processes*. Mar. 2000, pp. pp. 69–76, AAAI Press.

[21] D. Lenat, “Cyc: A large-scale investment in knowledge infrastructure.,” *Communications of the ACM*, vol. 38, no. 11, Nov. 1995.

[22] J. Grudin, “Groupware and social dynamics: eight challenges for developers,” *Communications of the ACM*, vol. 37, no. 1, pp. pp. 93–105, Jan. 1994.

[23] G. Attardi, A. Gull‘i, and F. Sebastiani, “Automatic web page categorization by link and context analysis,” in *THAI-99, 1st European Symposium on Telematics, Hypermedia and Artificial Intelligence*, C. Hutchison and G. Lanzarone, Eds., 1999, pp. p. 105–119.

[24] C. Faloutsos and D. Oard, “A survey of information retrieval and filtering methods,” Tech. Rep., Department of Computer Science, University of Maryland, College Park, MD, Aug. 1995.