# XAROP: A Midterm Report in Introducing a Decentralized Semantics-based Knowledge Sharing Application

Christoph Tempich[1], Marc Ehrig[1], Christiaan Fluit[2], Peter Haase[1], Esteve Lladó Martí[4], Michal Plechawski[3], and Steffen Staab[1]

[1]Institute AIFB, University of Karlsruhe, 76128 Karlsruhe, Germany
{ehrig,haase,staab,tempich}@aifb.uni-karlsruhe.de
[2] Aduna, Amersfoort, The Netherlands
Christiaan.Fluit@aduna.biz
[3] Empolis, Warsaw, Poland
Michal.Plechawski@empolis.pl
[4] Fundación IBIT
esteve@ibit.org
http://swap.semanticweb.org

**Abstract.** Knowledge management solutions relying on central repositories sometimes have not met expectations, since users often create knowledge ad-hoc using their individual vocabulary and using their own individual IT infrastructure (e.g., their laptop). To improve knowledge management for such decentralized and individualized knowledge work, it is necessary to, first, provide a corresponding decentralized IT infrastructure and to, second, deal with specific problems such as security and semantic heterogeneity. In this paper, we describe the technical peer-to-peer platform that we have built and summarize some of our experiences applying the platform in case study for coopetitioning organizations in the tourism sector.

## 1 Introduction

Knowledge management solutions relying on central repositories sometimes have not met expectations, since users often create knowledge ad-hoc using their individual vocabulary and using their own individual IT infrastructure (e.g., their laptop). [1] provide an explanation for the failed cases. They argue that traditional knowledge management systems take on a traditional managerial control paradigm, while subjectivity and sociality are essential features of knowledge creation and sharing. From that point they overhaul the traditional architecture of knowledge management systems towards a more decentralized architecture.

This general observation is supported by our own case study, which we pursued in the course of the SWAP project[1]. The case study is in the tourism domain of the Balearic Islands. The needs of the tourism industry there are best described by the term 'coopetition'. On the one hand the different organizations *compete* for customers against each

[1] http://swap.semanticweb.org/

other. On the other hand, they must *cooperate* in order to provide high quality for regional touristic issues like infrastructure, facilities, clean environment, or safety — that are critical for them to be able to compete against other tourism destinations. Working based on traditional, centralized knowledge management systems is infeasible, since no single organization can control all processes.

Although the need for decentralized knowledge management solutions is obvious, the field of solutions is still very limited. Two technologies are currently emerging as candidate solutions. On the communication level peer-to-peer (P2P) networks provide the means to connect the different participants (*cf.* [2]) while ontologies can provide the necessary expressivity on the representation level [3].

As for peer-to-peer networks one can distinguish three separate levels: *viz.* infrastructure, application and community level(*cf.* [4]). The P2P infrastructure provides basic mechanisms to communicate, security mechanisms, resource identification and peer identification. The P2P application provides services to the users supporting their process needs. P2P communities comprise the social activities which are enabled by the P2P paradigm.

Our use of ontology-based knowledge representation lies orthogonal to the three levels of the P2P network. The use of semantic descriptions of data sources stored by peers and indeed of semantic descriptions of peers themselves alleviates current problems on all three levels as we will show in the remainder.

In this paper we describe our experiences in introducing a peer-to-peer based knowledge management application. The application is built around and takes advantage of ontologies at the three different levels mentioned before. Before we start to explain this interaction, we briefly introduce the organizational setting of our case study (Section 2). From the case study we have derived several technical requirements (Section 3) for our application. The requirements led us first to an architecture for a semantically enriched peer-to-peer application (Section 4), and second to a number of new methods (Section 5). We finalize the paper with a summary of the case study (Section 6) and lessons learned (Section 6), a reference to the related work (Section 7) and the conclusion (Section 8).

## 2 Organizational Context

The case study we consider here is based in the tourism sector of the Balearic Islands. A number of organizations participating in the case study want to collaborate on some regional issues. Therefore they now collect and share information about *indicators* reflecting the impact of growing population and tourist fluxes in the islands, their environment and their infrastructures. Moreover, these indicators can be used to make predictions and help planning. For instance, organizations that require *Quality & Hospitality management* use the information to better plan, for example, their marketing campaigns. As another example, a governmental agency, a Balearic Government's coordination center of telematics, provides the local industry with information about *new technologies* that can help the tourism industry to better perform their tasks.

Due to the different working areas and objectives of the collaborating organizations, it proved impossible to set up a centralized knowledge management system or even a

completly centralized ontology. The case study partners asked explicitly for a system without a central server, where knowledge sharing is integrated into the normal work, but where very different kinds of information could be easily shared with others.

## 3 Requirements

From a technical point of view, the different organizations can be seen as one or many independently operating nodes within a "knowledge" network. Nodes can join or disconnect from the network at any moment and can live or act independently of the behavior of other nodes in the system. A node may perform several tasks. The most important one is that it acts as a peer in the network, so it can communicate with other nodes to achieve its goals. But apart from that it may act as an interface to interact with the human user of the network, or it may access knowledge sources to accomplish its tasks. One node may have one or more knowledge sources associated with it. These sources contain the information that a peer can make available to other peers. Examples are a user's filesystem and mail folders or a locally installed database.

A node must be designed to meet the following requirements that arise from the task of sharing information from the external sources with other peers:
*Integration:* Each piece of knowledge requires metadata about its origin. To retrieve external information, the metadata needs to capture information about where the piece of information was obtained from. This information will allow to identify a peer and locate resources in its repositories.
*Information heterogeneity:* As each peer may use its own local ontology, the distributed information is inherently heterogeneous. Mappings may be required, e.g. to overcome the heterogenous labelling of the same objects. However, in most cases some of the defined structures are very similar to each other. A general process is needed to identify commonalities and make them explicit.
*Security:* Some information may be of private nature and should not be visible to other peers. Other information may be restricted to a specific set of peers.
*Presentation:* In a peer-to-peer network queries are forwarded to different peers. Due to different network latencies and resources on the answering machines, answers can come at any time. Hence, the interface must help the user to distinguish between recent and old results, must update itself from time to time and should visualize where results come from.
*Network efficiency:* In peer-to-peer systems a general problem is, to distribute the queries in the network. Since, the number of messages increases exponentially with the number of hops a query is allowed to travel, intelligent query routing algorithms are required when the size of the network grows.

## 4 The SWAP platform

In the SWAP project, we have build the generic platform SWAPSTER to account for the general need of sharing semantic-based information in P2P fashion[2]. Based on SWAPSTER we have developed a semantic and P2P based knowledge management solution

[2] Bibster [5] is another solution for a different case

appropriate for the case study sketched above. The latter is called XAROP i.e. Catalan for syrup.

XAROP nodes wrap knowledge from their local sources (files, e-mails, etc.). Nodes ask for and retrieve knowledge from their peers. For communicating knowledge, XAROP transmits RDF structures [6], which are used to convey conceptual structures (e.g., the definition of what an indicator for airtravel is) as well as corresponding data (e.g., data about the number of arrivals by plane). For structured queries as well as for keyword queries, XAROP uses SeRQL, an SQL-like query language that allows for queries combining the conceptual and the data level and for returning newly constructed RDF-structures.

In the following we describe only the XAROP components that we refer to later in this document.

**Knowledge Sources:** Peers may have local sources of information such as the local file system, e-mail directories, local databases or bookmark lists. These local information sources represent the peer's body of knowledge as well as its basic vocabulary. These sources of information are the place where a peer can physically store information (documents, web pages) to be shared on the network.

**Knowledge Source Integrator:** The Knowledge Source Integrator is responsible for the extraction and integration of internal and external knowledge sources into the Local Node Repository. This task comprises (1) means to access local knowledge sources and extract an RDF(S) representation of the stored knowledge, (2) the selection of the RDF statements to be integrated into the Local Node Repository and (3) the annotation of the statements with metadata.

**Local Node Repository:** The local node repository stores all information and its meta information a peer wants to share with remote peers. It allows for query processing and view building. The repository is implemented on top of Sesame [7].

**User Interface:** The User Interface of the peer provides individual views on the information available in local sources as well as on information on the network. The views can be implemented using different visualization techniques (topic hierarchies, thematic maps, etc). One part of the user interface is the *Edit* component. The *Edit* component allows the user to supervise the mapping process and enables light weight ontology engineering.

**Communication Adapter:** This component is responsible for the network communication between peers. Our current implementation of the Communication Adapter is build on the JXTA framework [8].

**Information and Meta-information.** Information is represented as RDF(S) statements in the repository. The SWAP meta model[3] (*cf.* [9]) provides meta-information about the statements in the local node repository in order to memorize where the statements came from and other meta-information.

Besides the SWAP meta data model the SWAP environment builds on the SWAP common ontology[4]. The SWAP common model defines concepts for *e.g.* File and

[3] `http://swap.semanticweb.org/2003/01/swap-peer#`
[4] `http://swap.semanticweb.org/2003/01/swap-common#`

Folder. Purpose of these classes is to provide a common model for information usually found on a peer participating in a knowledge management network.

**Querying for Data.** SeRQL[10] is an SQL like RDF query language. The main feature of SeRQL is the ability to define structured output in terms of an RDF graph that does not necessarily coincide with the model that has been queried. This feature is essential for defining personalized views in the repository of a XAROP peer.

## 5 Method description on the three P2P levels

In this section we explain some of the methods that are most crucial to the XAROP knowledge management solution at the infrastructure, application or community level. Because of space restriction we skip over parts that are only of longer term interest, such as scalability towards a larger P2P network, which we have explore in another case study in somewhat more depth [5].

### 5.1 Infrastructure level: A distributed Security framework

Security considerations are of particular concern within a peer-to-peer framework. Users of a P2P system want to be sure that they give access to local resources only to trusted persons. Furthermore, they want to define different access control levels to their local knowledge, since they might trust different participants to various extent. While these consideration in themselves are difficult to handle, the security mechanism in a P2P system must be straightforward to define.

For authentication we use a public-key infrastructure (PKI) infrastructure with certificate authorities established within XAROP. A certain XAROP node acts as a root certificate authority for the XAROP system, all other peers will configure this node as trusted root certificate authority. In small networks, this certificate authority will issue certificates directly for users, whereas in large networks, it is possible to build a hierarchy of certificate authorities. The certificate creation will be done offline, on the configuration level. Certificates themselves will not be transmitted within the standard SWAP user interface.

The access control model has to be based on rules, since we cannot demand from users that they will enumerate privileged users for each local resource. The rules have to base on strict facts with proven origin (i.e. signed by the peer that generates a fact). A simple rule gives access for a single document (*e.g.* SWAP.doc) to a single, fixed person (*e.g.* Esteve). More complex rules are based on knowledge about both people and resources (All people involved in SWAP project can download all documents from my SWAP folder). All access control rights have to be explicit. Otherwise we assume that access is not allowed.

Further, the right to decide about the access properties of peers can be delegated to other peers. A person that we delegate the right to can be fixed or described by a similar pattern, forming a chain of trust (*e.g.* All people, about which Esteve said that they are involved in SWAP, and Mariusz said that they work for empolis, can download all my documents about SWAP). The access properties have to be digitally signed by the peer who assigned them.

**Example for IBIT** IBIT needs access control based on organization boundaries. On the other hand, it is not acceptable for an average user to be forced to create and maintain organizational information by herself. Using the described security model for the IBIT case an average user has to define (1) their administrator and (2) access control rules for his local resources. Administrators, define and maintain information about (1) organizational structures and (2) user membership to organizations.

### 5.2 Application level: Dealing with Semantic Heterogeneity and Visualization

**Heterogenity** Semantic mapping between ontologies is a necessary precondition to establish interoperation, i.e. overcoming the heterogeneity between peers using different ontologies. When we tried to apply existing tools [11, 12] to our scenario, we found that existing mapping methods were not suitable for the ontology integration task at hand. They have laid focus exclusively on improving the effectiveness and neglected efficiency, which becomes an issue already with the ontologies required in the context of Xarop. It is not sufficient to provide its user with the best possible mapping, it is also necessary to answer his queries within a few seconds - even if the two peers use two different ontologies and have never encountered each other before.
We briefly introduce the process that our approach follows [13]. It is started with two ontologies, which are going to be mapped onto one another, as its input. Mapping one ontology onto another means that for each entity (concept, relation, or instance) in ontology, we try to find a corresponding entity, which has the same intended meaning, in ontology. *Feature engineering* transforms the initial representation of ontologies into a format digestible for the similarity calculations such as RDFS. In a naive approach all entities of the first ontology are compared with all entities of the second ontology (*search steps*). However, in our approach we use heuristics to lower the number of candidate mappings. In specific, this means to compare pairs with similar labels or pairs for which neighboring entities have been assigned a mapping. The *similarity computation* measures are needed to compare the features of ontological entities. The features need to be extracted from extensional and intensional ontology definitions such as URIs, RDF/S primitives such as $subclass$, or domain specific features. Extremely costly features, in terms of runtime complexity, are replaced by less costly features. Then "String Similarity" measures the similarity of two strings on a scale from 0 to 1 based on Levenshtein's edit distance [14]; "SimSet" is used to compare sets of entities based on measures used for multidimensional scaling [15]. In general, there are several similarity values for a candidate pair of entities. These must be aggregated into a single aggregated similarity value. This is achieved through a summarization and normalization of adjusted similarity values. A similarity value above a certain threshold finally implies a mapping (*interpretation*). Several *iterations* of similarity calculations are needed to receive meaningful results. The returned output is a mapping table.
In first evaluation runs we have shown that our approach for identifying mappings between two ontologies is on a par with other good state-of-the-art algorithms concerning the quality of proposed mappings, while outperforming them with respect to efficiency. Using an approach combining many features to determine mappings clearly leads to significantly higher quality mappings. The here used approach is faster than standard

prominent approaches by a factor of 10 to 100. This makes the presented method a valid approach for the Xarop scenario.

**Visualization** We have developed the Cluster Map[16], for visualizing populated, lightweight ontologies as used for XAROP. It visualizes the instances of a number of selected classes from a hierarchy, organized by their classifications. Figure 1 shows an example Cluster Map, visualizing documents, classified according to topics discussed in those documents. The dark gray spheres represent ontology classes (the topics), with an attached label stating their name and cardinality. When a subclass relation holds between two classes, they are connected by a directed edge. The light yellow spheres represent instances. Balloon-shaped edges connect instances to the class(es) they belong to. Instances with the same class membership are grouped in clusters. Our example contains two clusters, one of them showing overlap between the two classes. Cluster Maps contain a lot of information about the instantiation of the classes, specifically exploiting the overlaps between them. For example, figure 1 shows that the "original lucerne" folder class has a significant overlap with the "swap idea". Such observations can trigger hypotheses about the available information and the domain in general. The graph layout algorithm used by the Cluster Map is a variant of the well-known family of spring embedder algorithms. Its outcome results in the geometric closeness of objects indicating their semantic closeness: classes that share instances are located near each other, and so are instances with the same or similar class memberships.

The Cluster Map is embedded in a highly interactive GUI, which is designed for browsing-oriented exploration of the populated ontology. Users can subsequently create visualizations of a number of classes by marking the check boxes in the class tree on the left pane. The software can animate the transition from one visualization to the next,

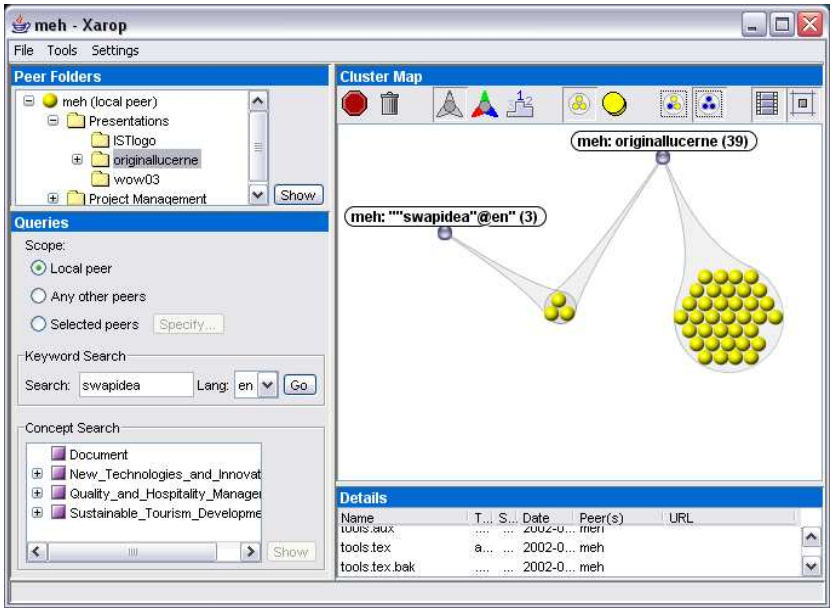


**Fig. 1.** Cluster Map

showing how the instances are regrouped in clusters. Through interaction a user can also retrieve information about the specific documents that are contained in a class or cluster. Further the visualization can be fine-tuned in several ways, in order to support certain tasks, or improve scalability. In the context of XAROP it was important to account for the particularities of P2P systems. Hence, the results are marked with the peer name they are coming from. Results are added to the Cluster Map incrementally, since not all peers answer at the same time. New results are highlighted. The search can be stopped when the user is satisfied. Thus it provides the usability requirements needed for the XAROP system.

### 5.3 Community level: The Distributed Ontology Engineering Process

Every participant in the XAROP network is allowed to structure his knowledge according to his needs. However, as we found in the case study, people working on the same issue have very similar ways of structuring information. Hence, a first step towards community building is to raise awareness about the existing commonalities within the group. From an ontological perspective that is equivalent to the agreement on a shared ontology. To enable the detection and building of shared ontologies we have defined a new ontology engineering process template *viz.* DILIGENT. It is important to note that the purpose of this process is not to agree on a conceptual model for the entire domain, but to find the subset of that model which is implicitly already agreed on. We introduce a board in charge of analyzing local ontologies and defining shared ones. This means that the participants can and should change the shared ontology after its publication. The DILIGENT process focuses in contrast to known ontology engineering methodologies available in the literature [17] on distributed ontology development involving different stakeholders, who have different purposes and needs and who usually are not at the same location.

We will now describe the general process, roles and functions in the DILIGENT process (cf. [18]). It comprises five main activities: (1) **build**, (2) **local adaptation**, (3) **analysis**, (4) **revision**, (5) **local update** (*cf.* figure 2). The process starts by having *domain experts*, *users*, *knowledge engineers* and *ontology engineers* **build**ing an initial ontology. The team involved in building the initial ontology, *viz.* the board, should be relatively small, in order to more easily find a small and consensual first version of the shared ontology. Moreover, we do not require completeness of the initial shared ontology with respect to the domain. On the first sight it seems contradictory that the case study partners do not want to share a common infrastructure but a shared ontology. However, the existence of a shared domain, overlapping or related competencies within organizations and the need for carrying out certain functions in cooperation suggests that it is possible to develop ontologies shared by the different sub-communities.
Once the product is made available, users can start using it and **locally adapting** it for their own purposes. Typically, due to new business requirements, or user and organization changes, their local ontologies evolve in a similar way as folder hierarchies in a file system. In their local environment they are free to change the reused shared ontology. However, they are not allowed to directly change the ontology shared by all users. Furthermore, the control board collects change requests to the shared ontology.
The board **analyzes** the local ontologies and the requests and tries to identify similar-

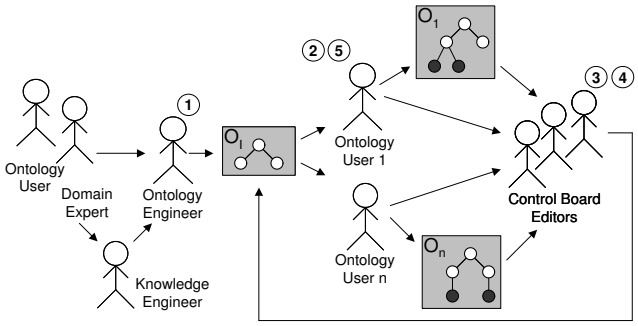


**Fig. 2.** Roles and functions in distributed ontology engineering

ities in the users' ontologies. Since not all of the changes introduced or requested by the users will be introduced,[5] a crucial activity of the board is deciding which changes are going to be introduced in the next version of the shared ontology. The input from users provides the necessary arguments to underline change requests. A balanced decision that takes into account the different needs of the users and meets user's evolving requirements[6] has to be found. The board should regularly **revise** the shared ontology, so that local ontologies do not diverge too far from the shared ontology. Therefore, the board should have a well-balanced and representative participation of the different kinds of participants involved in the process.

Once a new version of the shared ontology is released, users can **update** their own **local** ontologies to better use the knowledge represented in the new version. Even if the differences are small, users may rather reuse *e.g.* the new concepts instead of using their previously locally defined concepts that correspond to the new concepts represented in the new version.

## 6 The case study

More concretely four organizations including 21 peers took part in the case study and it is expected that the total number of organizations will grow by 7 to a total of 28 peers until September 2004.

**Early experiences** In a collaboration effort the KEEx [2] system was distributed among the participants in an early stage of the project. In the distributed version of KEEx users could share file, email and bookmark folders and the corresponding documents with the participants. Additionally to a keyword based query mechanism they could look at the contents of other participants folders. Furthermore they were able to provide manual mappings between their own folders and remote folders. This made it possible to query for remote documents by means of the local folder structure. From this early experiment we could draw two main conclusions which led us to our current solution. Our first conclusion was, that keyword based querying is not sufficient in the context

[5] The idea in this kind of development is not to merge all user ontologies.

[6] This is actually one of the trends in modern software engineering methodologies (see Rational Unified Process).

of the case study, since three different languages were in use and the search suffered from low precision. The performance of the search was improved by the use of a folder based approach. Our second conclusion was, that some users are willing to provided mappings between their structures and remote structures, but the additional effort to keep them updated is prohibitive.

**Current status** With XAROP users can share different kinds of folders and the corresponding documents. In the setup phase of the application users must define which information they want to share with whom. Additionally to the security infrastructure we introduced a small but shared ontology, to reduce the cost of mapping provision. Local documents can be associated automatically and manually with the concepts of the shared ontology. Participants can thus query for the content of others also with the concepts of the ontology. The shared ontology can be extended by the participants individually. The manual association task between documents and concepts was eased in two ways. Firstly, users could associate entire folder sub trees with certain concepts. Secondly, users could import their local folder structures and use the mapping tool to search automatically for correspondences between their local structures and the shared ontology. The mapping tool can also be used to search for correspondences between local and remote structures. When all participants had worked with the system for several days the board – comprising two domain experts and two ontology engineers – came together to analyze the changes made to the common ontology. The board could identify several concepts which are shared implicitly by all users. After some discussion the board decided to slightly change the shared ontology and redistribute a new version.

**Lessons learned** The case study helped us to better comprehend the use of ontologies in a peer-to-peer environment in general. The first lesson we learn is that security is the single most important issue in inter-organizational knowledge exchange. Without appropriate security mechanism users are not willing to share any information. Additionally to the implemented security mechanisms the participants would also like to know, who downloads which information. The possibility to grant access to certain documents on a 'per-query' base, was also requested. They could thus decide on query time which documents to share with whom.

On the application level we made the observation that the combination of keyword based and concept based search helps users to refine their query according to their information needs. At this point the ease of use of our user interface is very important. However, we must still work on the system and make it more light weight since not all participants work with the latest available computer equipment. Our mapping algorithm provides first suggestions for possible mappings very early, although these might be revised when the algorithm proceeds. This behavior was appreciated by most users.

Regarding the community level, the shared ontology was quickly accepted. Even so our users did understand the ontology mainly as a classification hierarchy for their documents and did not create instances of the defined concepts. Our expectation that the collaborative ontology engineering effort raised community awareness was met. Some of the participants changed their own folder structures in order to adhere to the commonly defined shared ontology. Other participants extended and changed it a bit. Later on this was the input for a refinement of the original ontology.

In spite of the technical challenges, user feedback was very positive since (i) the tool was integrated into their daily work environment and could be easily used and (ii) the tool provided very beneficial support to perform their tasks.

## 7 Related work

Knowledge management in Peer-to-Peer systems is the topic of various active research projects. Edutella [19] provides an RDF-based infrastructure for exchanging metadata in P2P applications. The Edutella Query Service is intended to be a standardized query exchange mechanism for RDF metadata stored in distributed RDF repositories. The Edutella project focuses on the education community. The Edamok project [2] also deals with distributed knowledge management in Peer-to-Peer systems and provides advanced facilities for mapping knowledge structures. However, neither of the systems is accompanied with an overall process for decentralized knowledge management on the community level. [20] presents a commercial P2P solution. They emphasize the organizational difficulties and the security concerns of the participants when introducing their P2P application. The system does not provide semantics-based representation of knowledge, and thus allows only for keyword-based queries.

## 8 Conclusion

In this paper we have described a solution for the recently recognized problem of decentralized knowledge management. We have developed a semantics based P2P knowledge sharing platform. In this platform, we introduced an appropriate security mechanism on infrastructure level. On the application level we introduced new mapping and visualization techniques. To foster community building we have defined a distributed ontology engineering process. We have described how the different methods have been applied in a concrete case study in the tourism domain. The results of our early experiments in introducing the application in the case study show the promises of our work. We could derive several lessons learned which will drive the future development of XAROP.

While we have addressed a broad range of problems in decentralized knowledge management, there are some issues to be addressed as part of future work. For example, query routing is currently not an issue within the XAROP system since the number of users is still small and simple query routing mechanism work effectively. However, in bigger networks, as they are found in the Bibster case study (the second system build on Swapster) efficient query routing is crucial. The interested reader may find some ideas of how to deal with such scalability issues in the description of the Bibster system[5].

*Acknowledgements.* Research reported in this paper has been partially financed by EU in the IST project SWAP (IST-2001-34103) and the IST project SEKT (IST-2003-506826). In particular we want to thank the other people in the SWAP team for their collaboration towards SWAPSTER.

## References


1. Bonifacio, M., Bouquet, P., Traverso, P.: Enabling distributed knowledge management: Managerial and technological implications. Novatica and Informatik/Informatique **III** (2002)
2. Bonifacio, M., Bouquet, P., Danieli, A., Donà, A., Mameli, G., Nori, M.: Keex: A peer-to-peer solution for distributed knowledge management. In Tochtermann, K., Maurer, H., eds.: Proceedings of the 4th International Conference on Knowledge Management (I-KNOW'04), Graz, Austria, Journal of Universal Computer Science (J.UCS) (2004) 43–52
3. D. OLeary: Using AI in knowledge management: Knowledge bases and ontologies. IEEE Intelligent Systems **13** (1998) 34–39
4. Schoder, D., Fischbach, K.: Peer-to-Peet Netzwerke für das Ressourcenmanagment. Wirtschaftsinformatik **45** (2003) 313–323
5. Haase, P., Broekstra, J., Ehrig, M., Menken, M., Mika, P., Plechawski, M., Pyszlak, P., Schnizler, B., Siebes, R., Staab, S., Tempich, C.: Bibster - a semantics-based bibliographic peer-to-peer system. In: Proc. of the 3rd Int. Semantic Web Conf. (ISWC 2004). (2004)
6. Klyne, G., Carroll, J.J.: Resource Description Framework (RDF): Concepts and abstract syntax. `http://www.w3.org/TR/rdf-concepts/` (2003)
7. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A generic architecture for storing and querying RDF and RDFSchema. In Horrocks, I., Hendler, J., eds.: Proc. of the 1st Int. Semantic Web Conf. (ISWC 2002), Springer (2002)
8. Gong, L.: Project jxta: A technology overview. Technical report, Sun Microsystems Inc. (2001)
9. Ehrig, M., Haase, P., van Harmelen, F., Siebes, R., Staab, S., Stuckenschmidt, H., Studer, R., Tempich, C.: The SWAP data and metadata model for semantics-based peer-to-peer systems. In: Proceedings of MATES-2003. First German Conference on Multiagent Technologies. LNAI, Erfurt, Germany, Springer (2003)
10. Broekstra, J.: SeRQL: Sesame RDF query language. In Ehrig, M., et al., eds.: SWAP Deliverable 3.2 Method Design. (2003) 55–68 `http://swap.semanticweb.org/public/Publications/swap-d3.2.pdf`.
11. Noy, N.F., Musen, M.A.: The PROMPT suite: interactive tools for ontology merging and mapping. International Journal of Human-Computer Studies **59** (2003) 983–1024
12. Doan, A., Domingos, P., Halevy, A.: Learning to match the schemas of data sources: A multistrategy approach. VLDB Journal **50** (2003) 279–301
13. Ehrig, M., Staab, S.: Qom - quick ontology mapping. In: Proceedings of the ISWC 2004, Hiroshima, Japan (2004)
14. Levenshtein, I.V.: Binary codes capable of correcting deletions, insertions, and reversals. Cybernetics and Control Theory (1966)
15. Cox, T., Cox, M.: Multidimensional Scaling. Chapman and Hall (1994)
16. Fluit, C., Sabou, M., van Harmelen, F.: Supporting user tasks through visualisation of lightweight ontologies. In Staab, S., Studer, R., eds.: Handbook on Ontologies in Information Systems, Springer-Verlag (2003)
17. Gómez-Pérez, A., Fernández-López, M., Corcho, O.: Ontological Engineering. Advanced Information and Knowlege Processing. Springer (2003)
18. Pinto, S., Staab, S., Sure, Y., Tempich, C.: OntoEdit empowering SWAP: a case study in supporting DIstributed, Loosely-controlled and evolvInG Engineering of oNTologies (DILIGENT). In Bussler, C., et al., eds.: 1st Euro. Semantic Web Symposium, ESWS 2004. (2004)
19. Nejdl, W., Wolf, B., Qu, C., Decker, S., Sintek, M., Naeve, A., Nilsson, M., Palmér, M., Risch, T.: EDUTELLA: A P2P networking infrastructure based on RDF. In: Proc. of the 2002 WWW Conference, Hawaii, USA (2002) 604–615
20. J. Schmücker and Wolfgang Müller: Praxiserfahrungen bei der Einführung dezentraler Wissensmanagement Lösungen. Wirtschaftsinformatik **45** (2003) 307–311