Increasing Knowledge Capturing Efficiency by Enterprise Portals

October 3, 2011

Purpose: Collaborative ontology editing tools enable distributed user groups to build and maintain ontologies. Enterprises that use these tools to simply capture knowledge for a given ontological structure face the following problems: (i) isolated software solution requiring its own user management, (ii) the user interface often does not provide a look-and-feel that is familiar to its users, (iii) additional security issues, (iv) hardly integrable into existing electronic work flows, and (v) additional deployment and training costs.

Design/methodology/approach: To address these problems, we designed, developed, and validated a plug-in concept for widely used enterprise content and collaboration portals. The prototype is implemented as a Microsoft SharePoint web part and has been validated in the risk and compliance management domain.

Findings: The research results enable enterprises to efficiently capture knowledge within given organizational and ontological structures. Considerable cost and time savings have been realized in the conducted case study.

Originality/value: According to our literature survey results, this work represents the first research effort that provides a generic approach to support and increase the efficiency of ontological knowledge capturing processes by enterprise portals.

Key words: knowledge capturing, enterprise portal, collaboration, ontology
Classification: Research paper

1 Introduction

Nowadays, excellent collaborative ontology editing tools such as Web Protege (Tudorache et al., 2008) are available and enable users to collaboratively build and maintain ontologies. Although these tools do not provide the full ontology engineering functionality of their rich client pendants, they enable the user to execute basic create/modify operations on classes, individuals, and properties. Conducting more complex ontology

engineering tasks can be supported by customizable plug-ins (e.g. widgets in the case of Web Protege).

However, at some point in the ontology engineering and maintenance process the structure of the ontology is stable and it is required to incorporate the actual knowledge into the structure. In most cases this requires users to create individuals[1] and to interrelate these individuals by the properties that were assigned to their classes. In the context of this paper we refer to this phase as *knowledge capturing*, i.e. different stakeholders are required to enter domain knowledge into a given ontological structure. Just as the stable ontological structure, this knowledge is crucial to finally use the ontology in its designated domain.

In the context of this paper we do not focus on supporting users at creating/modifying the structure of the ontology (i.e. introducing or modifying classes and properties). Instead, we explicitly support knowledge capturing as defined in the previous paragraph (i.e. adding, modifying, and deleting individuals).

Transferring the simplicity of Web 2.0 applications (i.e. enabling regular users to add and modify content) to ontology editing tools would enable regular users with no knowledge engineering background to extend and update knowledge in an ontology (cf. Braun et al. (2007)). Existing collaborative ontology editing software such as Web Protege already provides this simplicity to the user. However, enterprises may refrain from using existing collaborative ontology editing tools at knowledge capturing because they...

- introduce an isolated software solution requiring its own user management that can hardly be connected to the enterprise-wide user directory
- have a user interface look-and-feel that is different from the remaining software landscape
- introduce additional security issues and the need to patch an additional system and its libraries
- can hardly be integrated into existing electronic work flows
- generate additional deployment and training costs by requiring user training and the introduction of new infrastructure

To address these issues and to support enterprises at knowledge capturing, we developed a concept and prototypical implementation to connect enterprise content and collaboration portals with ontologies. Examples for enterprise content and collaboration portals are Microsoft SharePoint, IBM WebSphere, Oracle WebCenter, and SAP NetWeaver. Amongst others they provide the following functionality: collaboration tools, shared calenders, Blogs, Wikis, and document management. In this paper we use the term *enterprise portal* to refer to the aforementioned solutions. We review related work in Section 2, elaborate on the requirements in Section 3, and describe the overall

[1] In ontology engineering individuals are concrete instances/objects of classes

concept in Section 4. The prototypical implementation and the validation in the risk and compliance management domain is described in Section 5.

2 Related Work

This section briefly reviews existing work on supporting users in the knowledge capturing phase. Although knowledge capturing is an integral task at working with ontologies, only little research has been conducted at using enterprise portals for integrating knowledge into ontologies.

Norheim and Fjellheim (2006) developed a process-enabled knowledge management system to support offshore oilfield operations. The system captures knowledge gained in drilling operations and supplies relevant knowledge to planning new wells. The user interface of the system has been implemented using Mircosoft SharePoint. The RDF coded knowledge resources and reasoning engines[2] were hosted at the back end. Using an enterprise portal such as Microsoft SharePoint for the user interface supported the fast deployment and acceptance of the knowledge management solution. The shortcoming of this solution is its missing flexibility as it has been designed and implemented for a specific use case.

Braun et al. (2007) proposed an ontology maturing model which is based on the fact that ontology engineering is a collaborative informal learning process. It enables (i) capturing ideas from each individual, (ii) the consolidation in communities for a common terminology, (iii) creating formal lightweight ontologies, and finally (iv) axiomatization. One of the main goals of the model was to lower the barriers to ontology editing for non-knowledge engineers.

Baumeister et al. (2007) extended a standard Wiki system to allow for the capture, the maintenance and the use of knowledge systems. The system enables domain specialists to capture knowledge and to collaboratively build consultation systems. The Wiki system has been used to reduce the ontology engineering complexity.

Ding et al. (2008) developed a semantic content management solution based on an existing Microsoft SharePoint infrastructure. It enables users to tag documents and to execute semantic meaningful search queries. As the solution has been implemented in an already used enterprise portal, it enabled an efficient deployment in terms of user training and costs. The actual ontology and the bookmarking and mapping modules are implemented at the back end. As the system focuses on document tagging it cannot be used for knowledge capturing in terms of enriching ontologies with actual content.

Tudorache et al. (2008) developed Web Protege based on the widely used ontology editor Protege. Web Protege enables users to collaboratively build and maintain ontologies. The interface of Web Protege is fully customizable and enables developers to adapt the user interface to the users' needs.

Vasconcelos et al. (2009) developed an ontology-driven framework that incorporates expert annotations which integrate aspects of less tangible knowledge with more struc-

[2] A reasoning engine is a piece of software that infers logical consequences (new facts) from a set of asserted facts or axioms.

tured knowledge such as that stored in databases, procedures, manuals, and reports. The authors used Microsoft SharePoint to efficiently provide the functionality of their system to the end-users. The advantages of using an existing infrastructure for their deployment are (i) major development is only needed at the back end, and (ii) the actual user interface can be quickly deployed and provides a familiar look-and-feel.

Clark et al. (2001) developed a method to support users at knowledge capturing by viewing it primarily as a task of component assembly than axiom writing. These components are presented to the user as graphs, and the user can perform composition through graph manipulation operations. The overall goal of the authors was to create a knowledge capturing tool that does not require a trained knowledge engineer. Although, the work does not deal with enterprise portals, it clearly demonstrated the need for tools that enable users to simply add and modify knowledge in an ontology.

Alvaro et al. (2010) presented a concept of a semantic microblogging tool that can be used within enterprises as a lightweight method for knowledge management. It uses Web 2.0 concepts (similar to Twitter) to lower knowledge management entrance barriers. Messages sent by the users to the system are (i) semantically indexed, (ii) related to semantically related messages, and (iii) related to end-users (experts) that published the messages which have identified in Step (ii).

In sum, our related work survey revealed three insights: (i) there is a clear need for easily usable and integrated knowledge capturing tools, (ii) existing knowledge capturing support tools which are integrated in enterprise portals have been designed for specific problems and not as a general solution (therefore hardly reusable), and (iii) knowledge capturing capturing tools which do not rely on enterprise portals introduce the problems described in the Introduction section.

Our work addresses these problems and supplements past research in this area by providing a knowledge capturing approaches which is (i) integrated into existing enterprise portals and thereby providing a well known look and feel to the user, (ii) generically designed and therefore usable with every OWL ontology in any given problem domain, and (iii) fully integrated in the existing software landscape of the company and thereby not introducing any problems as described in the Introduction section (patching, access rights, work flows, deployment costs, etc.).

3 Enterprise Portals and Requirements

In this section we provide a brief enterprise portal market overview and state the requirements for our knowledge capturing tool.

Enterprise portals such as Microsoft SharePoint, IBM WebSphere, Oracle WebCenter, and SAP NetWeaver are widely used in medium- and large-sized enterprises. Microsoft, IBM, Oracle, and SAP dominated the enterprise portal market in 2009/2010 (cf. Murphy et al. (2010)). Enterprise portals increasingly aim at unifying user experience and provide relevancy and context across disparate applications (cf. Murphy et al. (2010)). The main idea is to have a single, personalized point of access to relevant information, processes and people.

Based on this trend, our main reasons for using enterprise portals for knowledge capturing are: (i) leveraging existing infrastructure, (ii) integration into existing work flows and access control models, (iii) providing a look-and-feel that is well known by the user, and (iv) low deployment and user training costs. The main objective of our research efforts is to provide organizations with an easily deployable tool to capture and manage their organization-specific knowledge in an existing ontological structure. To achieve this objective the following requirements have to be met by our knowledge capturing tool:

- Integration in productive enterprise portals to enable its integration in existing work flows and access control schemes
- Providing ontology modification functionality to read and write data from and to an ontology
- User interface that seamlessly integrates itself into the enterprise portal and provides a familiar look-and-feel to the user

Based on these requirements we developed a concept to utilize enterprise portals for ontology knowledge capturing.

4 Connecting Ontologies and Enterprise Portals

Figure 1 shows the overall concept of how we connect ontologies to enterprise portals[3].

The ontology and the ontology modification logic is hosted at the back end. The front end of our concept contains the enterprise portal including the knowledge capturing module and its user interface.

4.1 Back End

The back end provides portal-independent ontology modification methods to the front end. We use the OWL terms class, individual, and property to refer to ontology components (cf. W3C (2004)). The following functionality is implemented at the back end:

- Load and save the ontology [loadOntology(URI ontologyLocation), saveOntology(URI ontologyLocation)]
- Read individuals and subclasses of a given class [String[] getIndividuals(String class), String[] getSubClasses(String class)]
- Create and delete individuals in a given class [createIndividual(String individual, String class), deleteIndividual(String individual)]

[3]As the prototype has been implemented as Microsoft SharePoint web part we derived the server and service modules in Figure 1 from a MSDN SharePoint description (http://i.msdn.microsoft.com/dynimg/IC9815.gif). Further enterprise portals from IBM, Oracle, and SAP use similar functionality.

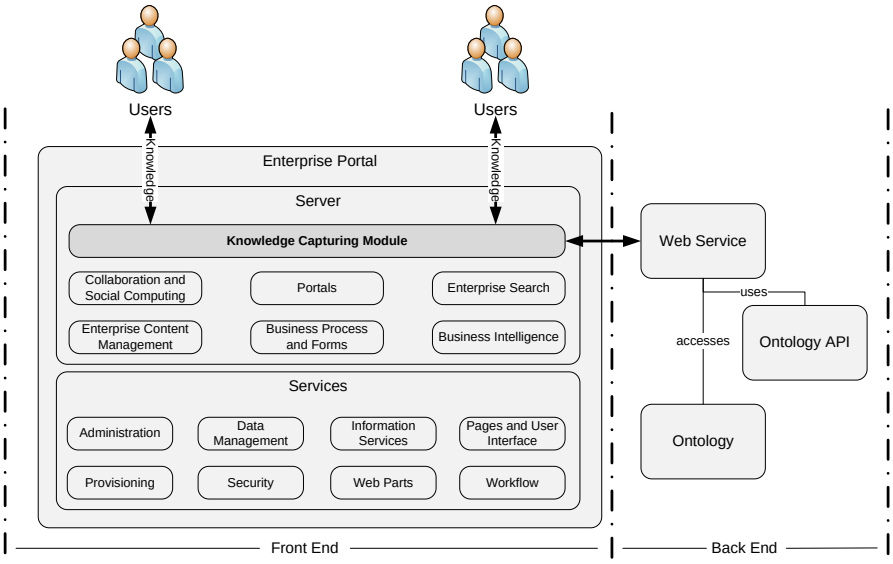

Figure 1: Architecture for connecting ontologies and enterprise portals

- Read property values of individuals [String[] getPropertyValues(String individual)]
- Write property values of individuals [writePropertyValues(String individual, String[] propertyValues)]

As the back end has been designed to support knowledge capturing in a given ontological structure, it does not allow the modification of the fundamental ontology structure (e.g. creating and deleting classes).

4.2 Front End

As depicted in Figure 1, enterprise portals already provide services such as work flows, security, data management, and general administration. On top of these general services, enterprise portals provide high level functions such as content management, business intelligence, and search. We use this entire service stack and integrate our knowledge capturing module as a separate module. The front end provides the following functionality:

- Connect to the web service and load/save the ontology
- Read and display the result sets of the web service methods *getIndividuals* and *getSubClasses* as a tree structure based on configurable root classes
- Display existing individuals and their property values (*getPropertyValues* method). The user interface is dynamically generated based on the result sets. E.g., a drop-down list shown for a property if it is a functional property defined by a certain range.

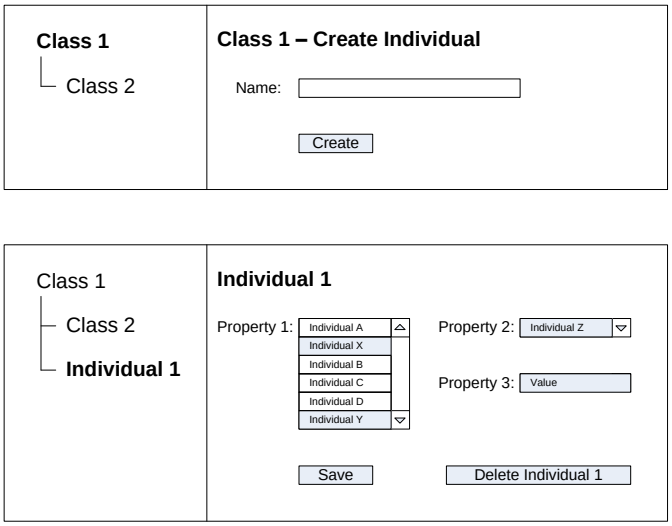

Figure 2: User interface concept

- At each individual, display only those properties that have been defined in the property white list. The white list is managed in the configuration file and ensures that only relevant properties are shown to the user.
- Delete individuals (*deleteIndividual*)
- Write individual data back to the ontology (*writePropertyValues*)
- Log all ontology operations with time stamp and user identification
- Restrict read and write access on class level

The configuration file of the front end defines (i) a set of root classes that should be displayed in the left-hand tree structure, (ii) enterprise portal user groups and their class read/write permissions, and (iii) white list that defines the properties that are displayed to the user.

Data access schema: A user is allowed to (i) change the properties of an individual if she/he has write permissions on the individual's class, (ii) read the properties of an individual if she/he is permitted to read its class, and (iii) create an individual if she/he has write permissions on the individual's class. User identification and management functionality is provided by the enterprise portal environment.

Figure 2 shows the basic user interface concept. The upper part shows the class view (*Class 1* is selected). Root classes such as *Class 1* on the left hand side are derived from the configuration file. By selecting a class the user can create a new individual of this class. The Name field specifies the label of the individual.

The bottom part of the figure shows the individual view (*Individual 1* is selected). By selecting an individual, the user can change its property values or delete the entire

individual. At non-functional object properties (e.g. *Property 1*) a list field enables the user to select appropriate individuals that fit into the property's range. At functional object properties (e.g. *Property 2*) a drop down field allows for specifying the property value. Text fields are used to capture non-functional and functional data properties (e.g. *Property 3*). At non-functional data properties we use semicolon separators to parse each value.

5 Case Study and Validation

The concept has been implemented as a Microsoft SharePoint[4] web part and has been validated in the information security risk and compliance management domain. For prototype development we use Microsoft SharePoint because of its high current and estimated future market share (cf. Radicati and Yamasaki (2010)). Although, no concrete market share numbers are freely available, Radicati and Yamasaki (2010) expect the installed base of Microsoft SharePoint to grow at an average annual rate of 11% from 2010 to 2014. Gartner reported that over 70% of its enterprise portal customer base use Microsoft SharePoint (cf. Murphy et al. (2010)). Amongst others Microsoft SharePoint provides enterprises with the following features: collaboration tools, shared calenders, Blogs, Wikis, and document management.

Based on the security ontology (cf. Fenz and Ekelhart (2009)), the AURUM tool (cf. Ekelhart et al. (2009)) has been developed to support enterprises at information security risk and compliance management. While general information security knowledge can be managed independently from organization-specific settings, risk and compliance management relies on accurate organization-specific information security knowledge to provide reliable risk figures. Examples for such organization-specific knowledge are (i) implemented security policies, (ii) type of malware scanner that is running on the main mail server, or (iii) business relevance of the file server. The main problems at capturing and integrating that type of knowledge are:

- Enable stakeholders with different IT literacy and geographic location to capture, check, and update the required knowledge
- Store the knowledge in a formalized and standardized format to enable automated knowledge processing
- Minimize the costs of managing this type of knowledge (e.g. deployment and training costs)

To address these problems, we implemented the prototype according to the concept described in Section 4. The back end (web service) uses the OWL API 3.1 to conduct

[4]Please note that Microsoft SharePoint refers to a whole family of products: (i) Windows SharePoint Services, (ii) Microsoft SharePoint Foundation, (iii) Microsoft Office SharePoint Server, and (iv) Microsoft SharePoint Server. In the context of this paper we refer to these products as Microsoft SharePoint.

the ontology modification operations as described in Section 4. The front end is implemented as a Microsoft SharePoint web part and is fully integrated into the existing portal solution of the company. The user interface of the front end is dynamically generated based on the ontological structure and already existing knowledge as described in Figure 2.

Case study methodology: The case study company was selected based on the following criteria: the company had to (i) already use an ontology-based system to evaluate potential improvements of our novel approach compared to the status quo, (ii) be open to new developments and wants to improve its knowledge capturing processes, and (iii) have an existing trust relationship to us as sensitive company data was involved in the case study. Based on these criteria we found and selected one of our partner companies which is active in the IT security area. The company employs about 20 people and has strict information security requirements because of several high volume non disclosure agreements with its customers. Management and four employees that are involved in the company's current knowledge capturing processes were involved in the case study. During the case study we observed and interviewed the employees regarding the usability of our novel knowledge capturing approach. At the end of the case study we interviewed management and involved employees regarding their opinion about potential benefits and limitations of our knowledge capturing approach.

The case study company uses the ontology-based AURUM tool to manage their information security risks. The first step at each risk management process is to inventory business-crucial resources (e.g. customer data) and assets that are capable of protecting these resources (e.g. a data backup policy). Based on this data, AURUM calculates the risk of the resource and provides options to decrease the risk (e.g. by the implementation of additional countermeasures). Traditionally the company assesses this knowledge by workshop-based methods and knowledge engineers incorporate it into the security ontology on which AURUM is based on. To remove the knowledge engineer dependency and to enable timely and decentralized knowledge capturing we deployed the prototype in the company's Microsoft SharePoint environment.

The management of the company defined four key employees that were responsible for capturing knowledge regarding business-crucial resources in their area of responsibility. These key employees were (i) one project manager, (ii) one facility manager, (iii) one secretary, and (iv) one IT administrator. Each of these persons was required to use our knowledge capturing tool to inventory their most important resources in terms of computers and data. For each resource they had to enter the following facts into the ontology:

- Properties of the resource (name, importance values, physical and virtual location)
- Existing countermeasures and their properties (name and their physical, virtual, and organizational location)

The security ontology classes *Computer*, *Software*, *Policy*, *Data*, and *TangibleAsset*

	Project manager	Facility manager	Secretary	IT administrator
Computer	Read	Read	Read	Read/Write
Software	Read/Write	Read	Read	Read/Write
Data	Read/Write	Read	Read/Write	Read/Write
Policy	Read/Write	Read/Write	Read/Write	Read/Write
Tangible Asset	Read	Read/Write	Read	Read

Table 1: Data access schema

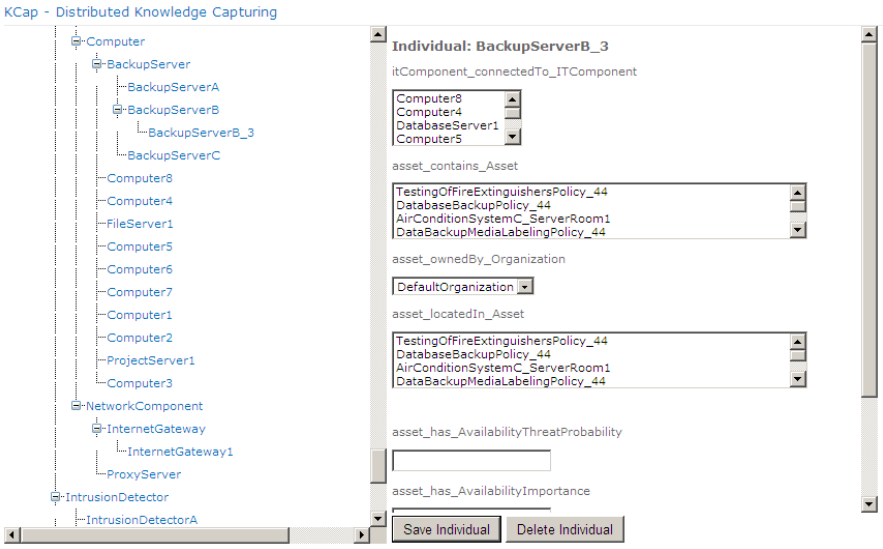

Figure 3: Enterprise portal knowledge capturing - user interface

have been defined as root classes. According to our data access schema each user is allowed to create individuals of classes on which she/he has write permissions on. Created individuals can be modified/read by users with write/read permission on the corresponding class, but only deleted by the individual creator. Table 1 shows the data access schema of our case study.

Figure 3 shows the user interface of our existing knowledge capturing prototype. Based on the security ontology we generate on the left hand side a tree structure that shows the available set of resource individuals and classes. By selecting one of these classes the user can create a new individual of this class. By selecting an individual, the right side of the user interface is dynamically populated with relevant individual properties and their values. Property values can be modified and entire individuals can be deleted by the user.

In the first knowledge capturing round, the facility manager used our prototype to establish a physical model of the company environment including the building itself, floors, rooms, doors, windows, access systems, alarm systems, fire detectors, and fire extinguishers. In the second round the IT administrator added IT assets such as com-

puters, software, and data to the ontology. The IT administrator also added existing policies (e.g. data backup policies) to the ontology and connected the IT assets by its properties to their physical and virtual location within the company (e.g. IT maintenance data is located on the file server, and the file server is located in the server room). In the third knowledge capturing round, the project manager and the secretary added business-crucial data resources and existing policies (e.g. visitor escort policy) to the ontology. The following items have been incorporated into the ontology (please note that the knowledge has been sequentially inserted as listed below):

1. Facility manager: one building, one floor, five rooms, five doors, five windows, two fire extinguishers
2. IT administrator: seven computers, nine installed software packages (seven virus scanners, one content filter software, and one spam filter software), three policies (data backup policy, secure password policy, and security update policy)
3. Project manager and secretary: three data items (customer data, project data, and finance data), four policies (confidential data encryption policy, visitor escort policy, key management policy, and locked doors policy)

Compared to the traditional workshop-based assessment, our prototype eliminated the need for a knowledge engineer, and enabled the users to enter, update, and check the knowledge on their own. Please note that this requires a stable base ontology such as the security ontology. Based on such a stable ontology, our tool enabled the users to add individuals to a defined set of root classes. The user interface only allows for basic individual operations and therefore does not require the support of a knowledge engineer. Further advantages of using the existing portal solution for knowledge capturing were: (i) no new front end IT infrastructure was required as our solution has been embedded in the existing enterprise portal (Microsoft SharePoint), (ii) existing user management systems (Microsoft Active Directory) were reused, (iii) users were familiar with the look-and-feel of the user interface, (iv) although, we have not implemented it, it would have been possible to build complex knowledge capturing work flows with the existing portal capabilities, and (v) low deployment and training costs. The following detailed case study results show the advantages and limitations of the developed approach:

Cost savings: In our case study, the deployment, the user training, and the actual knowledge capturing took a half working day and involved four employees of the company. In a traditional workshop-based setting, we estimate the required setup time to two days, including an external knowledge engineer that supports the four employees. The major advantage of using enterprise portals at knowledge capturing is that knowledge updates can be conducted by the end-user without the need of external help. In the context of our case study, up-to-date knowledge was required to ensure authentic risk and compliance management figures.

Quality of the created knowledge: The knowledge captured by our approach during the case study has been used in AURUM to check the validity of the captured knowledge. As the developed knowledge capturing tool directly operates on the ontological knowledge base, the captured knowledge was correctly incorporated into the ontology and risk/compliance figures regarding the added infrastructure components (computers, data, etc.) could be calculated by AURUM. The risk figures did not deviate from infrastructure components that were incorporated by traditional means into the ontology.

Security: As the knowledge encoded and processed in the case study involved sensitive and security relevant infrastructure, data protection against unauthorized read and write access is of utmost importance. As shown in Table 1 we elaborated on a data access schema that prevents unauthorized data access on the ontology's class level. The identifier and corresponding role of each user were retrieved from the enterprise portal which is connected to an company-wide user management system (Microsoft Active Directory).

Usability: The developed prototype reduced the interface necessary to capture relevant knowledge to an absolute minimum. Compared to traditional ontology editing tools it provides a user interface with a well known look and feel to the user. To achieve this reduced complexity users are only allowed to add/modify the ontology's individuals. Changes on the class level are not allowed. In the case study users appreciated the reduced complexity and did not feel that they had lost control over the modeled knowledge.

Disambiguation problems and conflict management: Especially, the project manager and the secretary which relied on knowledge created by the facility manager and the IT administrator (e.g., connecting the customer data to a specific computer to model its storage location), had disambiguation problems. As shown in Figure 3 the user is confronted with lists of existing individuals, but the meaning of each individual is not always clear (e.g., to which specific computer does the Computer8 individual refers to?). In our case study we addressed this disambiguation issue by requiring the users to describe newly created individuals by natural language descriptions. These descriptions are stored in comments attributes and support other users at relating the individuals to real world objects. As the case study implemented a straight forward sequential knowledge capturing process no unresolvable conflicts existed. However, conflict management tools such as discussion boards (provided out of the box by enterprise portals) would be required in different settings. In combination with discussion boards, moderators are required to finally approve and commit discussion results to the ontology (compare the approach used by Wikipedia).

During the validation the following limitations have been identified: (i) potential introduction of ontology inconsistencies due to asynchronous multi-user editing, and (ii) missing work flow definitions to support the knowledge capturing rounds (e.g. the rounds in our case study were (i) facility manager, (ii) IT administrator, and (iii) project manager and secretary. Each round is based on the output of the previous round). In further research we will address these limitations. Implementing locking strategies as proposed

by Seidenberg and Rector (2007) or using multi-user ontology editing libraries such as Collaborative Protege would be a potential solution to address the inconsistency problem. The second limitation can be addressed by utilizing work flow engines that are already present in most enterprise portal solutions. Implementing the knowledge capturing process as an electronic work flow is necessary to ensure that required knowledge is available at each knowledge capturing round.

The case study showed that the knowledge management effort of ontology-based systems can be reduced by using enterprise portals as user interfaces. The following reasons were identified during the case study: (i) the enterprise portal capabilities can be efficiently reused in the context of user interface design, user management, and work flow support, (ii) as the user interface is implemented as part of the enterprise portal users are already familiar with its look-and-feel, (iii) the simplified and restricted user interface prevents unauthorized knowledge modifications by restricting the allowed modification operations based on the role of the user, and (iv) conflict management can be supported by available functionality provided by the enterprise portal infrastructure (e.g. discussion boards).

6 Conclusion

One of the main problems at working with ontologies is enabling users to interact with the ontology. In the past years, substantial progress has been made by the development of collaborative ontology editing tools. However, some enterprise requirements such as usability, low deployment costs, and integration into existing software landscapes are not met by these tools. Therefore, we developed an approach to support enterprises at integrating knowledge into stable ontological structures by using already deployed enterprise content and collaboration portals. We validated the concept by a prototypical implementation (Microsoft SharePoint) that supports enterprises at knowledge capturing. In the conducted case study we reduced the time necessary for deployment, user training, and the actual knowledge capturing from two to a half working day. Furthermore, we eliminated the need for an external knowledge engineer and thereby enabled the company to instantly add new and modify existing knowledge. However, the validation revealed the following limitations of our approach: (i) potential ontology inconsistencies due to asynchronous multi-user editing, and (ii) missing work flow support. In further research we will address these limitations, improve the current prototype implementation, and conduct further real-world case studies. One possible hypothesis for further research is that the identified limitations can be addressed by technical means such as locking strategies and work flow engines.

References

Alvaro, G., Cordoba, C., Penela, V., Castagnone, M., Francesco Carbone, J. M. G.-P. and Contreras, J. (2010), Mikrow - an intra-enterprise semantic microblogging

tool as a micro-knowledge management solution, *in* 'Proceedings of the International Conference on Knowledge Management and Information Sharing', pp. 36–43.

Baumeister, J., Reutelshoefer, J. and Puppe, F. (2007), Knowwe: community-based knowledge capture with knowledge wikis, *in* 'Proceedings of the 4th international conference on Knowledge capture', K-CAP '07, ACM, New York, NY, USA, pp. 189–190.
URL: *http://doi.acm.org/10.1145/1298406.1298448*

Braun, S., Schmidt, A., Walter, A., Nagypal, G. and Zacharias, V. (2007), Ontology maturing: a collaborative web 2.0 approach to ontology engineering, *in* N. Noy, H. Alani, G. Stumme, P. Mika, Y. Sure and D. Vrandecic, eds, 'Proceedings of the Workshop on Social and Collaborative Construction of Structured Knowledge (CKC 2007) at the 16th International World Wide Web Conference (WWW2007) Banff, Canada, May 8, 2007', Vol. 273 of *CEUR Workshop Proceedings*.

Clark, P., Thompson, J., Barker, K., Porter, B., Chaudhri, V., Rodriguez, A., Thoméré, J., Mishra, S., Gil, Y., Hayes, P. and Reichherzer, T. (2001), Knowledge entry as the graphical assembly of components, *in* 'Proceedings of the 1st international conference on Knowledge capture', K-CAP '01, ACM, New York, NY, USA, pp. 22–29.
URL: *http://doi.acm.org/10.1145/500737.500745*

Ding, Y., Herzog, C., Luger, M., Prantner, K. and Yan, Z. (2008), Ontourism: semantic etourism portal, *in* 'Proceedings of 2nd International Scientific Conference of the e-Business Forum - E-Business in Travel, Tourism and Hospitality'.

Ekelhart, A., Fenz, S. and Neubauer, T. (2009), Aurum: A framework for supporting information security risk management, *in* 'Proceedings of the 42nd Hawaii International Conference on System Sciences, HICSS2009', IEEE Computer Society, Los Alamitos, CA, USA, pp. 1–10. 978-0-7695-3450-3.

Fenz, S. and Ekelhart, A. (2009), Formalizing information security knowledge, *in* 'Proceedings of the 4th ACM Symposium on Information, Computer, and Communications Security', ACM, New York, NY, USA, pp. 183–194. 978-1-60558-394-5.

Murphy, J., Phifer, G., Valdes, R. and Knipp, E. (2010), Magic quadrant for horizontal portals, Gartner RAS Core Research Note G00206214, Gartner.

Norheim, D. and Fjellheim, R. (2006), Aksio active knowledge management in the petroleum industry, *in* 'Proceedings of the ESWC 2006 Industry Forum'.

Radicati, S. and Yamasaki, T. (2010), Microsoft sharepoint market analysis, 2010-2014, Technical report, The Radicati Group, Inc.

Seidenberg, J. and Rector, A. (2007), A methodology for asynchronous multi-user editing of semantic web ontologies, *in* 'Proceedings of the 4th international conference on Knowledge capture', K-CAP '07, ACM, New York, NY, USA, pp. 127–134.
URL: *http://doi.acm.org/10.1145/1298406.1298430*

Tudorache, T., Vendetti, J. and Noy, N. (2008), Web-protege: A lightweight owl ontology editor for the web, *in* 'OWLED 2008'.

Vasconcelos, J., Kimble, C., Miranda, H. and Henriques, V. (2009), A knowledge-engine architecture for a competence management information system, *in* 'Proceedings of 14th UKAIS Conference'.

W3C (2004), 'OWL - web ontology language'.
URL: *http://www.w3.org/TR/2004/REC-owl-features-20040210/*