SemanticLIFE Collaboration: Security Requirements and Solutions – Security Aspects of Semantic Knowledge Management

Edgar R. Weippl, Alexander Schatten, Shuaib Karim, A Min Tjoa

Vienna University of Technology
<lastname>@ifs.tuwien.ac.at

Abstract.

SemanticLIFE is a project that stores all information an individual works with in a semantically enriched form. Ontologies are used to improve the search process and to express queries in the way humans think – e.g. "Find the draft I've been working on when traveling home from the conference in Chicago". When people cooperate on projects they obviously need to share information without spending time on entering keywords and thinking about who should be able to access which data; the issue is to correctly configure access controls so that only required information is shared with the appropriate people. Using a combination of the Chinese Wall and the Bell LaPadula model we show how access controls can be configured correctly with little effort by the users.

Introduction

The SemanticLIFE system is designed to store, manage and retrieve an individual's digital information accumulated over years. The system enables the acquisition and storage of data and creates semantic annotations to email messages, browsed web pages, phone calls, images, contacts, and other resources. SemanticLIFE also provides an intuitive and effective search mechanism based upon stored semantic knowledge [Ahmed].

While the first approach was to provide people with a tool to manage their personal (digital) life, we quickly recognized that individuals rarely work alone; instead, they permanently collaborate with others. Many emails and phone calls received by a person originate in the fact that the sender requires information. If the sender and the receiver work together on a regular basis, i.e. within a project, it would be more efficient if the information accumulated in the SemanticLIFE data base could be shared.

This paper presents security implications of allowing people to share their "Semantic Lives" (SemanticOFFICE); Two issues need to be addressed: (1) Access rights for new documents need to be set. (2) People join and leave projects teams or have informal contact with teams as expert advisors. Their access rights have to be dynamically managed. We thus show possible solutions to common issues of knowledge management in team processes: A combination of well-established access control models can be adapted to automatically assign access rights to new content. Users do not have to manually specify

who may access which file. Based on semantic annotations and content analysis access rights can be set similar to those of existing documents.

Related Work

In this area, a lot of work has already been carried out in some major projects. In this section we highlight their significant features and limitations.

MyLifeBits (Microsoft Research)

It is a system for storing all of one's lifetime data on a PC. The guiding principles are: (a) collections and search must replace hierarchy for organization, (b) multiple visualizations, (c) easy annotations (d) the authoring tool should support reuse of external references [Gemmel2]. As an experiment, G. Bell has captured all his articles, books, cards, etc, and stored them digitally. He is now paperless, and is beginning to capture phone calls, IM transcripts, television, and radio [Gemmel1]. They have successfully incorporated multiple annotation types, and creation of stories which are helpful for the short term memory. Facility for logical schemas modification and extension is at experimental stage.

They are still trying to explore features such as versioning, document similarity ranking and faceted classification. Until now, they were more concerned with functionality, but now the future work is related with UI, advanced visualization techniques, data mining for search, new capture mode and devices, shared usage, security, privacy and social issues.

Comparison with SemanticLIFE: Primarily it is perceived as a data store and not a retrieval box. The architecture is apparently compact and not open source. In contrast, the use of ontologies right from visualizations to storage, EDS support, and open source, make SemanticLIFE architecture more flexible, scalable and accessible. Currently, MyLifeBits does not offer any access control mechanisms beyond traditional access control lists (ACLs) for files and role-based access control (RBAC) for the database. It seems that "sharing of life experiences" has not yet been considered.

Haystack (MIT)

Haystack uses ontologies for information management [Huynh]. Its ultimate goal is to provide high-quality retrieval. Primarily it is designed as a single machine single user tool so as to give a psychological illusion of privacy and security. The guiding design principles are: (a) generic handling of all types of information, (b) flexibility to define additional information types by the user, (c) ability to define the interaction objects and associated operations directly by the user and (d) ability to delegate certain information processing tasks to the agents.

Haystack has a typical three tier architecture [Adar], i.e., a database layer, service layer and client layer. The focus of the database layer is more on augmenting its power

by personalizing off-the-shelf information retrieval and database tools to individual user needs and preferences. They have also done some implementation at the Trust layer of Semantic Web by using reification mechanism for RDF storage, and identifier strings as digital signatures during storage of RDF statements. The future developments include ontology conversion, enhanced query mechanisms using machine learning tools to improve retrieval, provide better interface for hybrid search, recommender system based upon user's interests derived from the user interaction, collaborative working, security, access control and privacy.

Haystack is an Open Source RDF-based information management environment, written in Java and is built into the Eclipse platform.

Comparison with SemanticLIFE: Haystack is primarily perceived as a system to provide improved user experience using standard semantic web technology. Its architecture is based upon RDF and the components are service oriented using Agents. Though it lacks support for EDS plugins at present, however it seems to be a strong, flexible and scalable architecture coherent with the Semantic web vision. Major emphasis is given on semantic UI and visualization and interaction ontologies which make its design stand out from others. They plan to implement the trust layer just beneath the UI and RDF store. Then it will be possible to manage the list of trusted users by the users themselves. Consequently only those RDF statements which are duly signed by these trusted users will provide information to UI, and not all. Haystack is implemented using a pull strategy. If the user selects an item available option are evaluated and presented. SemantifLIFE is a system that pushes the information into a semantic data store as soon as the data is available from the source system.

e-Person (HP)

Developed by HP, an ePerson is a personal representative on the net that is trusted by a user to store personal information, and make it available under appropriate controls for shared working environments. A test application 'SnippetManager' is developed, which allows users to manage collections of information items, annotate them and then share them with the community. HP's approach is focused on three principals, i.e., social filtering of information by the users themselves, structured knowledge in terms of ontologies mutually agreed upon by the communities, and person-centric instead of being corporate-centric in terms of ownership, vocabularies and scaling [Dave].

The ePerson infrastructure is designed as a series of layers, transport layer (TCP/UDP and Jabber transports), KB access layer (remote access to RDF stores and services), Structure layer (modeling of RDF vocabularies using DAML), Knowledge sources layer (provides specific knowledge services such as classification servers, importing profiles from the history server and a discovery server), and Applications layer (reusable UI components, viewing tools for knowledge based access during development and the SnippetManager application itself). Each of these provides an external API which abstracts the implementation details away from the client application. The testbed application SnippetManager, is based upon two guiding design principles. Firstly, to use RDF for storing all types of information, such as for the item metadata itself, the user's profile, the addressing and capability description of services, the user's preferences and configu-

rations and the current state of the application UI. This way, the same set of data manipulation and query tools can be used for exploiting every part of the architecture. Secondly, to clearly separate the details of messaging layer and physical hosting of services from the application code making free relocation of services across different networks and connection types.

They have successfully implemented a functional infrastructure for personal and shared information management, based upon semantic web techniques. ePerson project also highlights the fact that peer-to-peer knowledge sharing is different from other types, like peer-to-peer music sharing. In the later case, the information is readily usable. But in the first case, the contents of a knowledge worker's PC are not so much an archive as it is a workbench. A lot of this information is not readily sharable with others because it is still information under process [Jennifer].

Comparison with SemanticLIFE: ePerson is basically an information infrastructure in a shared environment. Unlike MyLifeBits, ePerson's focus is on sharing information. Therefore the work is highly relevant, especially the access control part and security implementation at the transport layer. A role based access control (RBAC) is used for hosting services,. Message communication at transport layer supports message signing and verification without using a full public key infrastructure (PKI), authority, authentication and end-to-end security. But it also creates problems when there are slight changes at transport layer message format. Statement level access control is provided for RDF store. Roles associated with allowable information patters, are assigned to trusted users.

Lifestreams (Yale University)

Lifestreams [Eric], is an academic project from Yale University. It is a personal store that uses a simple organizational metaphor, a time-ordered stream of documents combined with several powerful operators that replaces many conventional computer constructs (such as named files, directories, and explicit storage). Their work on the client side includes an x windows client, command line interface and a PDA client.

The motivating ideas were, (a) storage should be transparent, (b) directories are inadequate as an organizing device, (c) archiving should be automatic, (d) the system should provide sophisticated logic for summarizing or compressing a large group of related documents on one screen, (e) reminding should be made more convenient, and (f) personal data should be accessible anywhere and compatibility should be automatic. Consequently, they implemented a model incorporating document creation and storage, directories on demand, overviews and chronology as a storage model.

While their current prototype is for managing personal information, in future a "lifestream" could also be used as a natural framework for managing enterprise information and web sites. Other future directions include shared usage, UI design, system integration, indexing and retrieval, usage of agents, network access, security, and performance issues.

Comparison with SemanticLIFE: In our opinion the system provides a good visualization of your lifetime information items and is a good substitute for the desktop metaphor. The project is perceived more from UI point of view; thus issues in storing and analyzing data storage using semantic information is not their main focus.

Edutella (The SUN project JXTA)

Edutella [Nejdl] is an RDF based metadata infrastructure for P2P applications. It is a useful project for getting an insight into shared usage of SemanticLIFE project. Its service oriented architecture supports querying RDF metadata, persistent data storage, mapping between different set of metadata vocabularies, and annotation services.

Edutella is based upon JXTA, an open source project with a layered architecture (core, services and applications layer) and XML based protocols to cover typically P2P functionality. This layered approach fits nicely for the Edutella infrastructure. Applications like repositories, annotation tools and GUI interfaces connected to and accessing the Edutalla network are implemented in the Application layer. Exchange of query, query results and other metadata between Edutella peers is carried out at Services layer. Peers register the query at the services layer by specifying the supported metadata in a standardized way. Queries are sent through Edutella network to the subset of peers who have registered with the service. For this purpose QEL (Query Exchange Language) is defined. Queries can be expressed in RDF form, and thus can be visualized by a query graph. Data persistency, availability and work load balancing is achieved by replicating metadata in additional peers. Mapping between RDF schemas at different locations is provided in such a way that queries across peers are also translated accordingly.

Comparison with SemanticLIFE: The comprehensive query language, mediation and registration services for P2P access can be very beneficial for interconnecting SemanticLIFE between different people. The added functionalities of cascaded analysis of information items, knowledge refinement in ontologies, aggregation, ranking of query results, interactive visualizations and accessibility for people with special needs clearly distinguish our project. Edutella uses the underlying transport layer security (TLS) which is common to JXTA framework. Secure communication between the peers is done by public key cryptography.

Semagix Freedom (Ex. Protégé)

Semagix Freedom architecture which is nicely tiered between schema, business logic and the use of extractor agents for knowledge source data access is based upon ontologies. These provide automated categorization, cataloguing and extraction of semantically enhanced metadata from content sources [Sheth2]. Agents can extract content from web sites and file repositories, as well as trusted data from unstructured (news articles), semi structured (web sites, Intranets) and structured sources (databases, spreadsheets etc). Knowledge Agents extract trusted data and resolve ambiguities before inserting or merging data into the ontology. Content Agents retrieve data for extraction of semantic metadata. The Semantic Enhancement Server classifies aggregated content into the appropriate topic/category, and subsequently performs entity extraction and content enhancement with semantic metadata from the Freedom ontology. Metabase index resides in main memory all the time, to accelerate the retrieval process. Adapters are implemented for exchange of metadata with other applications. Semantic query server enables users to make search in text, execute ambiguous queries, retrieve relevant and personalized content that is actionable and in context. On top of all this, semantic visualization is provided for easy and intuitive exploration [Sheth1].

Applications can be implemented in a modular, phased manner to allow incremental scaling. Open APIs provide external access to a rich set of ontology and metabase features. The approach is successfully used in different domains such as airline passenger risk assessment for airports, anti-money laundering solutions and market segmentation and resource allocation model for the music industry. Based on an XML orientated architecture, support of multiple metadata standards and Web Services is provided. It also provides automatic document format conversion to XML, to facilitate metadata extraction. A flexible scheduling mechanism is provided to take into account any database changes dynamically.

Comparison with SemanticLIFE: The architecture is designed for corporate content management systems; it covers corporate content only. SemanticLIFE stores an individual's entire digital life and makes the relevant parts available to coworkers.

Security Aspects of Semantic Knowledge Management

As mentioned before, the SemanticLIFE project captures an individual's activities while working with her computer, PDA and cell phone. The sophisticated mechanisms to retrieve information empower users to work much more efficiently.

However, today's knowledge workers usually work in teams. Team work can only be efficient if sharing of knowledge works efficiently, too. Therefore the knowledge stored in the SemanticLIFE database should be accessible by coworkers. The obvious security issue resulting from this feature is that coworkers should not have access to an individual's entire digital life, for three reasons:

1. People often work on more than one project and coworkers should only have access to data that is related to a project in which they participate.
2. Within a certain project, some information should not be freely distributed. For instance, notes indicating that the project manager intends to terminate an employee should remain confidential – or, another example, the whistle blower who uncovers illegal transactions needs to collect confidential data to prove the accusations.
3. Even when using a company's equipment, people will communicate privately, e.g. receive private phone calls.

Secrecy Requirements

The first step in every security risk analysis is to identify the assets that need to be protected. Looking at the data stored in the SemanticLIFE database two separate entities can be distinguished: (1) the ontology and (2) the instances.

In this paper we only focus on secrecy requirements of instances. We do neither consider security implications of sharing the ontology definitions nor other security requirements such as integrity or availability. The mechanism used to implement secrecy requirements is access control. Access control is used to allow only authenticated users perform authorized operations on data item.

Hierarchical Organization of Projects

We thus need to identify who should be granted access to data under which conditions. Users working on the same project should be able to share data; however, based on their job description, they should not be able to access all data. Most projects are organized hierarchically.

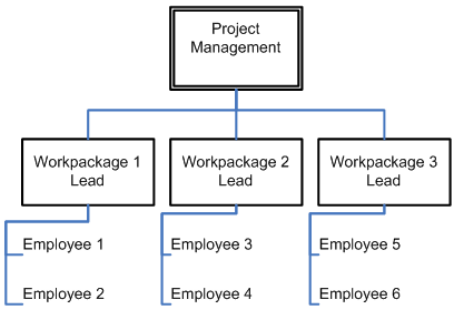

Fig. 1. Hierarchical Project Structure

In multi-level security (MLS) systems, data is categorized according to its level of secrecy, e.g. into 'project management', 'work package lead', 'unclassified'. Users then receive clearance to access data on certain levels. Thus, users are authorized to view all documents found on their own level or below. In this way it can be guaranteed that users cannot access information that does not correspond to their level. Write access, however, is only permitted on the same level. That is to say, a user who is authorized to access 'secret' data, may produce documents exclusively on the 'secret' level (during one session). In this way, users are prevented from (mistakenly) copying 'secret' data into a 'public' file. The above-mentioned security model was developed by Bell LaPadula (BLP). The levels are ordered so that

1. 'unclassified' ≤ 'work package lead'
2. 'project management' ≤ 'work package lead'

A simplified version of the ss-property (a.k.a no-read-up property) states that a user S can read a data object O if and only if $L_O \leq L_S$.

Conflict of Interest

The Chinese Wall model (CW) [Brewer] implements both confidentiality and integrity. It primarily addresses the issue of conflict of interest: if a consultancy works for two competing clients (company A and company B), consultants should not be able to transfer information from one client to its rival (see Fig 2).

We follow Bishop's notation [Bishop] that a company dataset (CD) stores data objects (O) related to a single company and a conflict of interest (COI) contains all CDs pertaining to competing companies. PR(S) is the set of objects that a user S has read.

CW-simple security property: A user (S) can read O only if either one of the following two conditions is true:

1. ∃O': S has previously accessed O' and CD(O') = CD(O).

2. $\forall O'$: $O' \in PR(S) \Rightarrow COI(O') \neq COI(O)$.

This policy works fine; however, it is too strict since it assumes that all data of a company must not be used by the competitor. By introducing a third condition this can be relaxed:

3. O is a data object that does not contain sensitive data (i.e. sanitized object).

Looking at write operations, the user S can write a data object O if following two conditions are both met:

1. S is granted read access on O by the previous three conditions
2. For all sanitized objects O', S can read $O' \Rightarrow CD(O') = CD(O)$.

Comparison of BLP and Chinese Wall Model

The two models are different because in the BLP model users have associated clearance labels whereas in the CW model they do not. The BLP model is a static model that does not take prior access into account. In the CW model, however, the set PR(S) changes over time and reflects the history of object accesses by a user S.

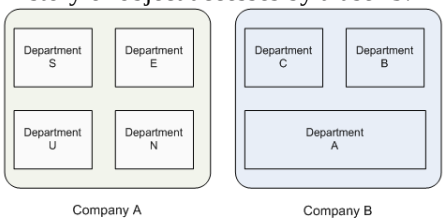

Fig. 2. Classes showing conflicts of interests (COIs) [Bishop]

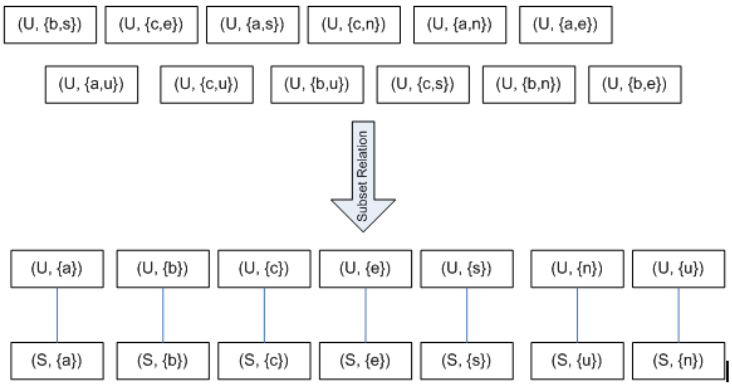

Fig. 3. BLP levels and compartments can be used to emulate a Chinese Wall model at any given (static) state [Bishop].

The BLP model can be used to emulate the CW model [Bishop] in a given state. A security category is assigned to each (COI, CD) pair and two security levels are defined: S =

sanitized, U = unsanitized. S dominates U. However, the modification of the access rights over time that characterize the CW model cannot be expressed in the BLP model.

As previously shown, the CW model and the BLP model represent different requirements that are both relevant to sharing semantically enriched information between co-workers: the CW model can be used to implement the dynamic component and the BLP model the hierarchical approach.

Reference Monitor

According to Bishop [Bishop] "a *reference monitor* is an access control concept of an abstract machine that mediates all accesses to objects by subjects. A r*eference validation mechanism* (RVM) is an implementation of the reference monitor concept. An RVM must be tamperproof, must always be invoked and can never be bypassed, and must be small enough to be subject to analysis and testing, the completeness of which can be assured." (p 502).

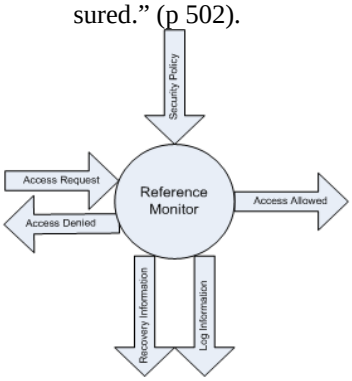

Fig. 4. Reference monitor [Povey].

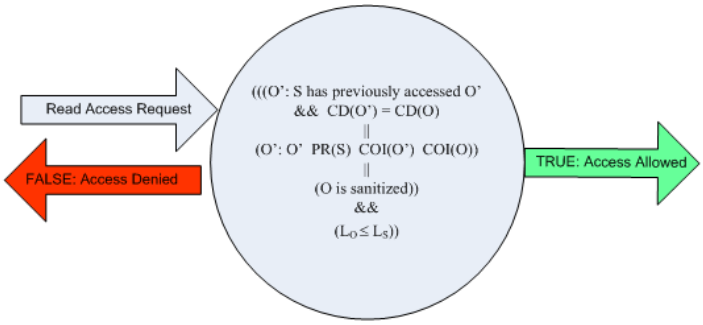

Fig 5. Evaluation of the proposed security policy.

We propose to use the CW model's CW-simple security property to check in a first step whether a user may access data or whether a conflict of interest occurs. If the CW-simple security property would grant the user access, a second check is performed. The BLP model's ss-property is used to evaluate whether a user should be granted access. A user is permitted access to data object only if both the CW-simple security property and the ss-property grant access (Fig 5).

Automatic Classification

For the aforementioned system to work in a real-world scenario two major issues have to be addressed:

1. How can the system know which data object is sanitized and if it is not sanitized which project it relates to.
2. How can the system downgrade data objects within the BLP model?

Determining the state of sanitation

If a new data object is created by a user, the system first needs to determine whether the data is sensitive or not. Based on a small set of rules, a decision can be made for many data objects. For instance, if a new email is received, a couple of rules (Fig 6) are evaluated.

Domain	Rule	Result	Type
Email	If receiver = "all@company.com"	Sanitized	Company-wide rule
Email	If sender = "susan"	Not sanitized (private)	Individual rule
Email	If body contains <cell phone number>	Not sanitized (<project group sender> ∩ <project group receiver>)	Individual rule

Fig. 6. Rules to determine conflicts of interests

The first rule in Fig 6 states that if an email is sent to the company-wide mailing list it does not contain sensitive information for one or more projects. The second rule indicates that all emails received from Susan are considered private emails. The third rule is used to classify all emails that contain a cell phone number as sensitive messages open only to the members of the project groups that both the sender and the receiver are members of.

The next step is that the system looks for semantically similar data objects that are already classified. It then takes the greatest lower bound (g.l.b.) of restrictions – for sets of projects the g.l.b. is the intersection of the sets. The rationale is that we follow the "least privilege" principle and thus select the most restricting rule.

Once the system has found a suitable classification it is advisable to prompt the user and ask for confirmation. Even though some users might consider this annoying at first, it makes the system much more trustworthy and trust is of paramount importance for user acceptance. Just imagine that the system decides to share your private emails with the whole company simply because you have a new girlfriend and her emails do not yet match a rule and are too different from emails sent by your previous girlfriend for the matching to work reliably.

Setting BLP levels

After establishing which project a data item is related to, we need to determine the level (Project Management (PM) – Work package Lead (WL) or unclassified) a data object pertains to. Weippl [Weippl] proposes to use a SOM-based approach to find similar data objects and to classify the new item accordingly. This process can be improved by taking the underlying ontology into account. We expect that this approach will work especially well in our setting because all texts pertain to the same project and the SOM will thus not separate according to projects but to other differences in the content.

Integrity Checking

The integrity and "correctness" of the ontological model is an important issue in most applications of the semantic office. As ontologies can become arbitrary complex, with no formal boundary conditions, a complete integrity check is not possible automatically. However the SOM-based classification algorithm (as suggested in the previous section) can be used to give the user hints about possible inconsistencies in the model.

Consider the following example: users build an ontology/taxonomy, where two categories (A and A') are actually semantically similar. This can happen as models grow and continuously undergo refactoring processes. As new objects are entered into the system, the SOM algorithm will suggest for certain objects the category A *and* A'. If such "double or multiple categorizations" occur repeated, the system would suggest the user, that a potential simplification of the model could be possible.

Actor and organisational models for a SemanticOFFICE – a knowledge management approach

Access control is a tedious task in most enterprise systems. Often access control lists or similar mechanisms are used to control which user/user group might access which resources. Often this type of model is too simple and moreover is not a direct representation of the "business reality".

As our SemanticOFFICE should capture individual as well as organization wide information objects, and the system information management is based on semantic web standards, it stands to reason to capture also the access control logic in a semantic way.

First of all, we want to avoid, keeping parallel models of the enterprise: a "business model" and a "access control model". The same is true for the actual information objects. Hence the first step is to build an ontology that captures the structure of the business (organization), essentially the actors involved. In the second step, the business processed and activities are modeled and expressed in an ontology schema.

The main idea now is this: the connection between the actor ontology and the process/activity ontology can be formally described and can be used to determine access control logic. To give a concrete example:

An enterprise may have (among other activities) projects and processes. The first step is, to define an ontology for these types of activities. E.g., a project has a project manager, who is a person. A project consists of tasks. Each task may have subtasks. Tasks may be dependent. Each task has attributes like start date, end date, man hours and so on. Each project must be authorized by a board of directors and the finance department and so on. The same has to be done for the process specification. A process has a process owner, has tasks, and so on…

The next step is to define the ontology for actors. This is essentially a map of the companies' hierarchy, but not only. Customers are actors, associate and business partners are actors; systems as web services might be actors and so on. As a matter of fact, all persons, organizational units or systems that have an active or passive role in a business activity have to be taken into consideration.

As soon as these models are prepared, the connection, the glue between these two has to be defined. E.g., one could define, that a person who is director has read access to all

objects that are defined in the activity ontology except to such, that are private property of some other person.

A customer might use specific web services, whereas a associate partner may use other web services to modify data. A particular web service itself might have access to specific data objects according to the definition.

Up to now, these definitions are merely schemes, templates. Every actual entity must be an instance of one of these categories defined. As soon as this categorization is done, the access control is defined by implicit rules. No further definition of access control lists is required! E.g., a person who is assigned as project manager of a specific project automatically has the appropriate access rights to, e.g., read, write and modify all objects in the projects.

New information objects that entered into the system are now categorized by two boundary conditions: first of all, following the ontological rules according to the person or system who/which enters the object, and secondly by the automatically suggested category following the SOM algorithm.

Still we have to face two problems: First of all, the business and actor ontologies themselves are subject of continuous change. This has significant consequences: an object which was accessible to a specific person might become inaccessible according to a changed policy. Several options are conceivable: The policy change might only be applied for objects that are entered after the change, or the policy change might apply to all available objects. Most probably, this will be an option that has to be defined in changing the policy.

The second more general issue is the fact that many access policies will not be clear in defining the business and actor ontologies, or might depend on the concrete boundary conditions. These boundaries typically change over time, which is addressed by the CW model. Hence the logic that defines the implicit access control has to have rules of different types: first of all strict rules will be necessary like: “a project manager may change basic attributes of all project tasks”. Then also weak rules must be possible such as: “Depending on the information object, the project manager may decide whether a specific object is accessible by a specific group or not”. Here the SOM categorization comes again into play. The SOM can find similar documents and suggest to use the previous documents’ rules for the new document.

Conclusion

Information secrecy is crucial for building the trust layer for a shared SemanticOFFICE. Combining easy-to-understand models allows users to really understand what content they share. Even though models such as the BLP may seem a little outdated they have the obvious advantage that they are easy to understand. In many cases the model will be implemented using RBAC controls since RBAC is supported by all major databases systems. The next step is to address additional requirements such as integrity and availability and to expand the level of protection from the instances to the ontology itself.

References

[Ahmed] Ahmed M., Hoang H.H., Karim M.S., Khusro S., Lanzenberger M., Latif K., Michlmayr E., Mustofa K., Nguyen H.T, Rauber A.,Schatten A., Tho M. N. and Tjoa A M, 'SemanticLIFE' - A Framework for Managing Information of A Human Lifetime, The Sixth International Conference Information Integration and Web-based Applications and Services., 27-29th Sep, 2004, Jakarta-Indonesia.

[Adar] Eytan Adar, David Karger, and Lynn Andrea Stein, Haystack: Per-user information environments, Conference on Information and Knowledge Management., 1999.

[Bishop] Matt Bishop. *Computer Security: Art and Science*. Addison-Wesley-Longman, 2002.

[Brewer] D. Brewer and M. Nash, The Chinese Wall Security Policy, Proc. Of the 1989 IEEE Symposium on Security and Privacy pp 206-214, May 1989

[Dave] Banks Dave, Cayzer Steve, Dickinson Ian, and Reynolds Dave, The ePerson snippet manager: a semantic web application, Tech. report, HP Laboratories Bristol, HPL-2002-328.

[Eric] Freeman Eric and Gelernter David, Lifestreams: A storage model for personal data, ACM SIGMOD Record, Bulletin 25,1, March 1996, pp. 80–86.

[Gemmel1] Jim Gemmel, Gordon Bell, and Roger Lueder, Mylifebits: Living with a lifetime store, ATR Workshop on Ubiquitous Experience Media, Sept. 9-10 2003.

[Gemmel2] Jim Gemmel, Gordon Bell, Roger Lueder, Steven Drucker, and Curtis Wong, Mylifebits: Fulfilling the Memex vision, ACM Multimedia ’02, December 2002, pp. 235–238.

[Huynh] David Huynh, David Karger, and Dennis Quan, Haystack: A platform for creating, organizing and visualizing information using RDF, Semantic Web Workshop, 2002.

[Jennifer] Hyams Jennifer and Sellen Abigail, Gathering and sharing web-based information: Implications for ”ePerson” concepts, Tech. report, HP Laboratories Bristol, HPL-2003-19.

[Nejdl] Wolfgang Nejdl, Boris Wolf, Changtao Qu, Stefan Decker, Michael Sintek, Ambjorn Naeve, Mikael Nilsson, Matthias Palmer, and Tore Risch, Edutella: A P2P networking infrastructure based on RDF, WWW, ACM, May 2002.

[Povey] Dean Povey, Enforcing Well-formed and Partially-formed Transactions for Unix, Proceedings of the 8th USENIX Security Symposium, August 23-36, 1999, Washington, D.C. pages 47-62

[Quan] Dennis Quan and David Karger, How to make a semantic web browser, Proceedings of the 13th Conference on World Wide Web, May 17-20 2004.

[Sheth1] Amith Sheth and David Avant, Semantic visualization: Interfaces for exploring and exploiting ontology, knowledgebases, heterogeneous content and complex relationships, NASA Virtual Iron Bird Workshop, CA (2003).

[Sheth2] Amith Sheth and Cartic Ramakrishnan, Semantic (web) technology in action: Ontology driven information systems for search, integration and analysis, IEEE Data Engineering Bulletin, Special issue on Making the Semantic Web Real (2003).

[Weippl] Edgar Weippl, Ismail Khalil Ibrahim, and Werner Winiwarter. Content-based management of document access control. In The Proceedings of the 14th International Conference on Applications of Prolog, pages 78-86. Prolog Association of Japan, November 2001.