International Journal of Knowledge Management

January-March 2013, Vol. 9, No. 1

Table of Contents

Research Articles

1 **ICT and Building a Knowledge-Based Society in Egypt**
Nagla Rizk, Access to Knowledge for Development Center, School of Business, The American University in Cairo, Cairo, Egypt
Sherif Kamel, School of Business, The American University in Cairo, Cairo, Egypt

21 **Virtual Communities as Narrative Networks: Developing a Model of Knowledge Creation for Crowdsourced Environments**
Michael R. Weeks, Department of Management, Sykes College of Business, The University of Tampa, Tampa, FL, USA
Natasha F. Veltri, Department of Information & Technology Management, Sykes College of Business, The University of Tampa, Tampa, FL, USA

42 **Search Efforts, Selective Appropriation, and the Usefulness of New Knowledge: Evidence from a Comparison Across U.S. and Non-U.S. Patent Applicants**
Osamu Suzuki, Institute of Business and Accounting, Kwansei Gakuin University, Nishinomiya, Japan

60 **Integrating Knowledge Sources: An Ontological Approach**
Hayden Wimmer, Department of Information Systems, University of Maryland Baltimore County, Baltimore, MD, USA
Victoria Yoon, Department of Information Systems, School of Business, Virginia Commonwealth University, Richmond, VA, USA
Roy Rada, Department Of Information Systems, University of Maryland Baltimore County, Baltimore, MD, USA

76 **A Generic Approach for the Semantic Annotation of Conceptual Models Using a Service-Oriented Architecture**
Hans-Georg Fill, Stanford Center for Biomedical Informatics Research, Stanford University, Stanford, CA, USA, & Department of Knowledge and Business Engineering, University of Vienna, Vienna, Austria
Daniela Schremser, Department of Knowledge and Business Engineering, University of Vienna, Vienna, Austria
Dimitris Karagiannis, Department of Knowledge and Business Engineering, University of Vienna, Vienna, Austria

IJKM is listed or indexed in the following: ACM Digital Library; Bacon's Media Directory; Burrelle's Media Directory; Cabell's Directories; Compendex (Elsevier Engineering Index); CSA Illumina; DBLP; DEST Register of Refereed Journals; Gale Directory of Publications & Broadcast Media; GetCited; Google Scholar; INSPEC; JournalTOCs; KnowledgeBoard; Library & Information Science Abstracts (LISA); MediaFinder; Norwegian Social Science Data Services (NSD); PsycINFO®; SCOPUS; The Index of Information Systems Journals; The Standard Periodical Directory; Ulrich's Periodicals Directory

A Generic Approach for the Semantic Annotation of Conceptual Models Using a Service-Oriented Architecture

Hans-Georg Fill, Stanford Center for Biomedical Informatics Research, Stanford University, Stanford, CA, USA, & Department of Knowledge and Business Engineering, University of Vienna, Vienna, Austria

Daniela Schremser, Department of Knowledge and Business Engineering, University of Vienna, Vienna, Austria

Dimitris Karagiannis, Department of Knowledge and Business Engineering, University of Vienna, Vienna, Austria

ABSTRACT

In this paper a generic service oriented architecture for the semantic annotation of conceptual models is described. It allows to annotate elements of conceptual models with concepts from formal semantic schemata. Thereby, additional semantic functionalities for models can be realized. Due to the integration of aspects of service orientation, the platform can be easily extended to support different modeling and semantic schema languages. Furthermore, it can act as an integration platform for other tools working on models and ontologies.

Keywords: *Computer Science, Information Systems, Integration Platform, Semantic Schemata, Semantic Technologies, Service Oriented Architecture*

INTRODUCTION

The capturing of knowledge using conceptual models has been discussed in several areas such as strategic and business process management, software engineering or enterprise architecture management cf. (Borgida, Chaudhri, Giorgini, & Yu, 2009; Fill, 2009). Thereby, conceptual models are used to formally describe aspects of the physical and social world for the purposes of understanding and communication (Mylopoulos, 1992). They are based on a formal syntactic definition that requires human interpretation in order to be processed. To leverage the semantics contained in these models to a machine processable level, the annotation of conceptual models using formal semantic schemata has been proposed (Lautenbacher, Bauer, & Seitz, 2008; Höfferer, 2007). With these annotations additional functionalities can be provided to the

DOI: 10.4018/jkm.2013010105

users of conceptual models. Examples include mechanisms to achieve the interoperability of different modeling languages (Höfferer, 2007), to measure the similarity of model instances (Ehrig, Koschmider, & Oberweis, 2007), to prepare process models for the further transformation to executable workflows (Lautenbacher & Bauer, 2007) or to identify dependencies on compliance requirements (Fill & Reischl, 2009).

When implementing these approaches in IT tools two directions can be taken: Either the modeling tools are extended to support the annotation functionality and thus preserve the modeling environment that the user is familiar with (Born et al., 2008; Fill & Burzynski, 2009). Or, completely new modeling tools are designed. However, in both cases the annotation functionalities and the modeling tools are tightly coupled. Furthermore, the re-use of the functionalities for different types of modeling languages and formal semantic schemata is often limited due to this close integration.

To overcome these deficiencies we will describe an approach that allows for the combination of modeling components for arbitrary modeling and formal semantic schema languages to support semantic annotation. The approach is based on the concepts of service orientation. This permits to achieve a greater flexibility and adaptability in regard to modeling languages, semantic schema languages, and other modeling tools. The remainder of the paper is structured as follows: In the section "Foundations" we will outline the foundations used for our approach, in the section "Semantic Annotation Architecture" the conceived architecture is presented. In the section "Implementation using ADOxx® and Protégé" the concrete implementation of the approach is discussed. The paper is concluded with an outlook on the future work.

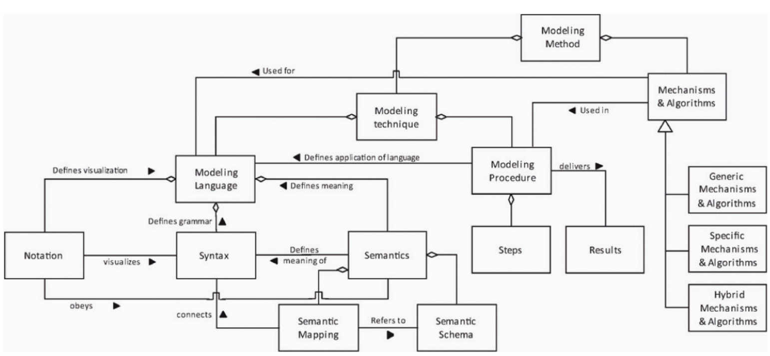
FOUNDATIONS

In this section we will give a short introduction to modeling methods and conceptual models, semantic annotations and the concepts of service orientation that are necessary to describe our approach.

Modeling Methods and Conceptual Models

To ensure a common understanding of the terms in regard to modeling methods and modeling languages we will refer in the following to a framework from the area of meta modeling that has been developed in (Karagiannis & Kühn, 2002). In this framework, the central components of a modeling method are a modeling technique and mechanisms and algorithms (see Figure 1). The modeling technique consists of a modeling language that is defined by its syntax, semantics, and notation. The notation is thereby separated from the syntax to allow for an independent specification of the visual representation of the modeling language (Fill, 2009). The modeling procedure defines the

Figure 1. Components of modeling methods (Karagiannis & Kühn, 2002)



steps that can be applied to the modeling language to create results. The mechanisms and algorithms can either be generic, specific or hybrid and are used by the modeling procedure. Generic mechanisms and algorithms can be applied to all modeling languages, whereas specific mechanisms and algorithms are only applicable to a particular modeling language. In the hybrid case, mechanisms and algorithms are basically generic but can be specifically adapted to a modeling language, for example to improve usability (Karagiannis & Kühn, 2002). The representation of the modeling language can itself be represented as a model which is then commonly denoted as a meta model cf. (Höfferer, 2007). This meta model provides the abstract syntax of the modeling language that can be instantiated to a concrete syntax in the form of models.

Conceptual models are specified by using a formal syntax together with human interpretable semantics. This allows to exactly describe the modeling elements and their relations and implement the modeling language in IT-based model editors (Fill, 2004). However, as conceptual models are intended to be used by humans and not machines, the semantics of the modeling elements are usually not formally defined (Mylopoulos, 1992). This applies both to the type semantics, i.e. the meaning of the abstract syntactic elements and their relations, as well as to the inherent semantics, i.e. the meaning assigned to the elements of the concrete syntax in the resulting models.

Semantic Annotation of Conceptual Models

In order to raise the semantics contained in conceptual models to a machine processable level, the syntactic elements can be annotated with concepts from a formal semantic schema. Thereby, the particular advantage of conceptual models in regard to the intuitive understanding by users who are not familiar with mathematical or logical expressions can be preserved. At the same time, additional semantic processing functionalities can be provided for conceptual models.

Semantic schemata permit to define words and meanings of a knowledge area (Obrst, 2003). Thus, they provide a machine processable representation of semantic relationships that can be used to perform automated reasoning. With the annotations, linkages between the syntactic elements of the modeling language and the elements from the semantic schema are established. On the side of conceptual models these linkages can either be defined on the meta model level, i.e. by referring to the abstract syntax of the modeling language, or on the model level, i.e. by referring to the concrete syntax cf. (Lautenbacher et al., 2008). For the definition of the semantic schemata various languages are available. Examples for semantic schema languages that have been successfully used for annotations comprise logic-based languages such as OWL and RDF (Fill & Burzynski, 2009; Thomas, 2007), thesauri (Höfferer, 2007) or controlled vocabularies (Fill & Reischl, 2009). In the following we will use the term ontology to refer to all types of formal semantic schemata.

Similar to the concepts for storing meta data, semantic annotations can either be stored as (a.) additions to the abstract or concrete syntax definitions of the modeling language, (b.) by adding reference attributes to the abstract or concrete syntax definitions, or (c.) by using a separate annotation specification that does not require to modify any of the modeling or ontology languages (Abramowicz, Filipowska, Kaczmarek, & Kaczmarek, 2007; Kiryakov, Popov, Terziev, Manov, & Ognyanoff, 2004). Based on these annotations the semantic processing of conceptual models can be enhanced by using the additional information from the referenced ontologies. Examples for the concrete application of annotations include semantic searches for model content including corresponding visualizations (Fill & Reischl, 2009), the integration of models or the translation between different modeling languages (Höfferer, 2007).

The benefit of semantic annotations is thereby achieved through the ability to process the model content using semantic phenomena such as synonymy, sub/superordination, semantic similarity or logical inference. When searching for example for an activity in a conceptual process model named "Issue statement" a traditional search would only return activities that syntactically match the words in the search string. Through semantic annotation, the activity may also be related to concepts in an ontology. In this way the activity may be related for example to the synonym group containing "to issue" in the WordNet® ontology[1] . By accessing the information of the synonym group the query can thus be expanded to search for terms such as "to publish", "to bring out" and "to release". A similar annotation may have been put in place for an activity named "Print statement". Again, the WordNet® synonym group of "to print" can be assigned. This synonym group also encompasses the term "to publish" from which a direct relation to the synonym group of "to issue" can be established. Thus, an expanded search algorithm can take into account this information and retrieve activities that not only match syntactically but also ones which are semantically similar. Additional ways of processing may also take into account the semantic similarity based on hypernym/hyponym relationships or additional logic specifications, e.g. in the form of rules.

Concepts of Service Orientation

As has been mentioned above, the realization of semantic annotation functionalities often leads to a tight coupling of the involved components. However, for developing semantic annotation functionalities that can be used in practical scenarios, the components may need to be adapted to specific user requirements. This applies in particular to the use of different modeling and ontology languages as well as to the distributed access to information contained in the models and ontologies. To overcome these drawbacks and make a system easily extensible and adaptable to new requirements, it can be reverted to concepts of service orientation and service oriented architectures (SOA) (Erl, 2005; Singh & Huhns, 2005). As there are currently several definitions for SOA available in the literature, we will restrict ourselves to the following principles which stand for commonly used aspects of service orientation: According to (Erl, 2005) services should be loosely coupled, re-usable, composable, stateless, autonomous, discoverable, share a formal contract, and abstract the underlying logic. It is important that the services are independent of the service consumer, other services and the architecture, so that the service is loosely coupled and autonomous as well as stateless. Therefore, it is possible and desired to re-use the service, because the services share a formal contract which contains all input and output data as well as all exceptions of the service. Another essential aspect is that services can be easily composed with others so that the services have to be discoverable by service consumers. Thereby, the services act like black boxes that hide the underlying logic. Hence, the system is heterogeneous and dynamic (Singh & Huhns, 2005). For realizing these concepts a large number of technologies and standards are available, including the web of services standards of the W3C[2] .

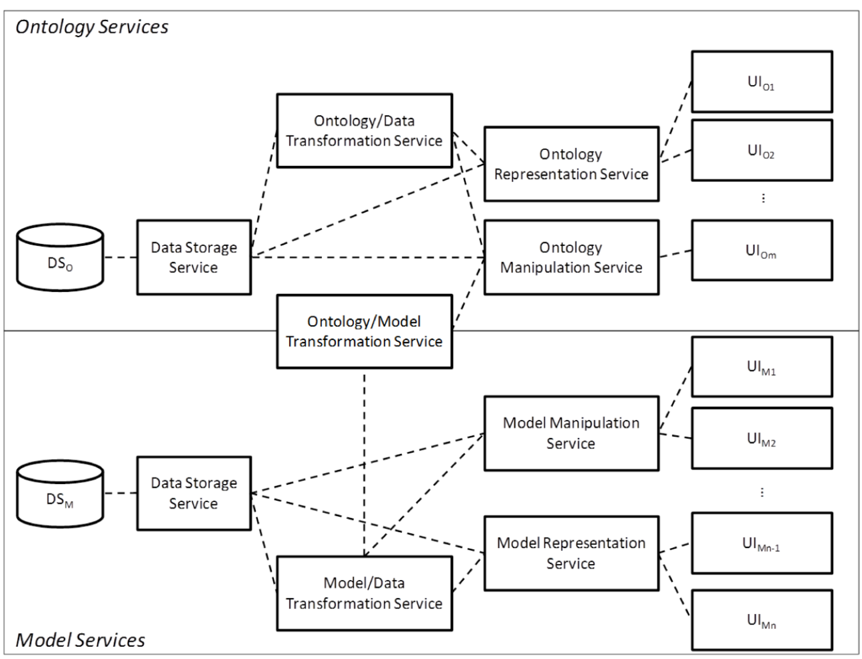
SEMANTIC ANNOTATION ARCHITECTURE

Based on these foundations the concrete requirements for a semantic annotation architecture can now be derived. From a conceptual point of view models and ontologies share several characteristics (Atkinson, Gutheil, & Kiko, 2006) that can even be used to realize the integration of ontologies as models (Fill & Burzynski, 2009). However, the nature of conceptual models differs substantially from that of ontologies. As already mentioned above, conceptual models are intended to be used by humans and not machines. This is reflected by several aspects, which directly influence the according implementations. For example, a single conceptual model rarely contains as many elements and relations as an ontology due to the fact that human users still need to be able to "process" its contents. An ontology on the other hand may benefit from containing several thousand elements in order to conduct useful inferencing. In addition, as ontologies are usually expressed using logic based languages (Obrst, 2003), the corresponding editors for ontologies need to consider the formal mathematical relationships and explicitly support the user in defining correct statements. In contrast, conceptual models provide a higher degree of freedom to the user and do not require to obey formal semantic relationships.

Although it can be largely debated whether these differences allow for a separation of conceptual models and ontologies, it is a fact that tools used for conceptual modeling and tools for creating and maintaining ontologies have hitherto been separately developed. To make them available for tasks such as semantic annotation therefore requires an architecture that is capable of maintaining the integrity of the single tools while at the same time providing appropriate interfaces to achieve their interoperability. These considerations have led us to the conception of a semantic annotation architecture that is based on the concepts of service orientation.

In Figure 2 a generic architecture is shown that allows to semantically annotate conceptual models with concepts from an ontology. Based on the separate development of ontology and modeling tools, the architecture is divided into two parts. The upper part represents services working on ontologies and the lower part those working on models. An underlying assumption of this architecture is that ontologies can be represented as models using a meta modeling approach as has been described in (Fill & Burzynski, 2009). Thereby, the syntax of an ontology language is represented using a visual notation as it is commonly done for models. The resulting visual language can then be used to visualize and manipulate ontologies in the same way as models. To ensure efficiency and scalability, the approach is based on a three-tier architecture to separate data storage, the main parts of processing logic and the user interfaces. It contains two data storages, one for ontology information (DSO) and one for model information (DSM). Both storages are accessible only through corresponding data storage services that encapsulate the technical storage operations. They are also responsible for the user management to control which user

Figure 2. Generic architecture



or service is allowed to perform read or write operations.

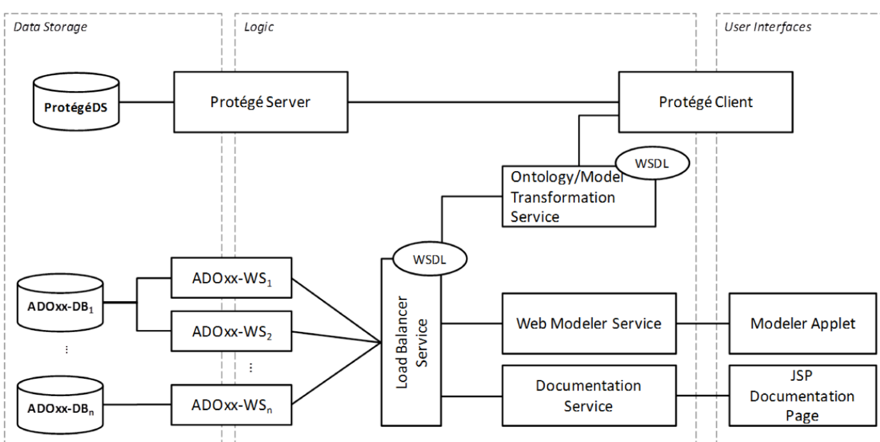
The conversions between ontology data and model data to the data formats required by the data storage are handled by corresponding transformation services. Although model or ontology information may also be directly storable (e.g. by using a triple store[3]) the use of other storages such as relational databases requires a conversion. Ontologies and models are manipulated using distinct manipulation services. These are used to create, modify, and delete models and ontologies and act as interfaces to specific user interfaces. They may also directly access the data storage services to store and retrieve user information and permissions. With representation services only a read access to ontology or model information is possible, again depending on the permissions of the user accessing the service. The actual semantic annotation is supported through the ontology/model transformation service (Figure 3). This service mediates between the ontology manipulation service and the model/data transformation service. It thus permits the ontology manipulation service to transform an ontology into a model and make it accessible to the model manipulation service by using the model/data transformation service. In case the model manipulation service is generic enough to work on arbitrary modeling languages - as it is e.g. in the case for meta modeling platforms (Karagiannis & Kühn, 2002) - no adaptation effort is required to support the semantic annotation. The architecture is complemented by a range of user interfaces. These are either used for accessing ontology information ($UI_{O1..m}$) or model information ($UI_{M1..n}$).

It is further assumed that for every service in the architecture a machine readable service description is available in a central repository that provides information about the service operations, its endpoints and its input and output messages. Thereby, the services are made discoverable, re-usable, and composable. Besides the service description most of the services shall also be implemented in a way that the loose coupling is achieved.

IMPLEMENTATION USING ADOXX® AND PROTÉGÉ

To validate the conceived architecture it has been implemented in the context of the Open Models Initiative (OMI). The initiative "intends to establish a community of people who focus on the creation, maintenance, modification, distribution, and analysis of models" (Karagiannis, Grossmann, & Höfferer, 2008, p.3). Besides its community aspects it provides a

Figure 3. Semantic annotation architecture based on ADOxx® and Protégé



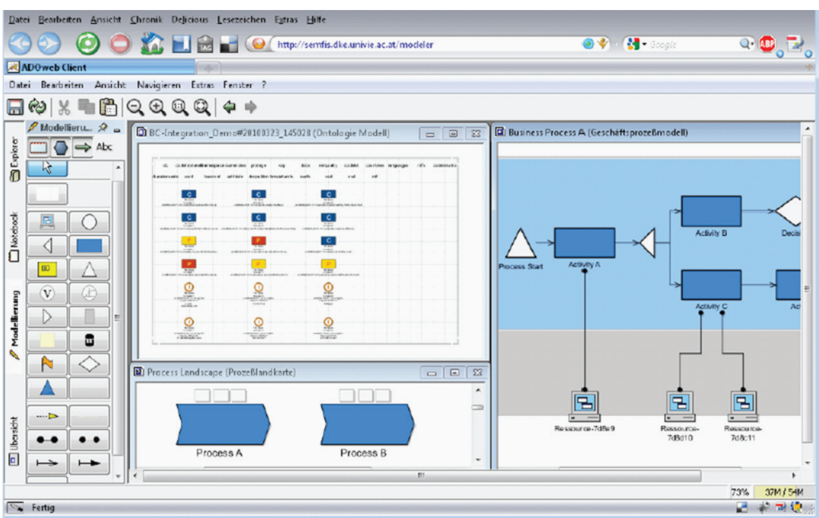
set of tools to implement conceptual visual modeling languages. These tools are based on the ADOxx® platform that allows for the definition, storage and processing of meta models and the automatic creation of model editors based on these meta models. ADOxx® has been used to develop a number of commercial modeling tools including the business process management suite ADONIS®, the performance management toolkit ADOscore®, the supply chain toolkit ADOlog® and the IT infrastructure management toolkit ADOit®[4]. ADOxx® contains several components that enable an industry-adequate handling of models such as advanced user and user group management, highly scalable model repositories with database storage facilities, and elaborated analysis and simulation functionalities. Besides that it comprises a set of web service interfaces, which made it an optimal choice for realizing our intended service oriented architecture.

For the handling of ontology related aspects we chose the widely used Protégé tool which is an open source tool developed by Stanford Center for Biomedical Informatics Research[5] (Musen, 1989). It provides support for a variety of ontology languages including the common web ontology language OWL, RDFS, and RDF. Due to its widespread use many users are familiar with the interface and functionality of Protégé. Its flexible plugin architecture allowed us to easily implement the service interfaces required for our approach (Figure 4). Furthermore, it offers several storage functionalities including interfaces to triple stores and relational databases.

The concrete architecture for the semantic annotation is depicted in Figure 3. The data storage is divided in a data storage for Protégé, which could be files, relational databases or triple stores, and relational databases for ADOxx®. The role of the data storage services in the generic architecture are taken over by the Protégé server and a number of ADOxx® web services (ADOxx®-WS$_{1..n}$). Every ADOxx®-WS contains a particular meta model that is stored in a corresponding database together with a set of models based on this meta model. In addition, the ADOxx®-WS are also responsible for the handling of user permissions on the models. The load balancer service distributes the requests to the different ADOxx®-WS.

In a similar way, the Protégé server handles the storage of ontologies and ontology instances and the user management for ontologies. Via the ontology/model transformation service a

Figure 4. Screenshot of the ADOxx® web modeler applet user interface



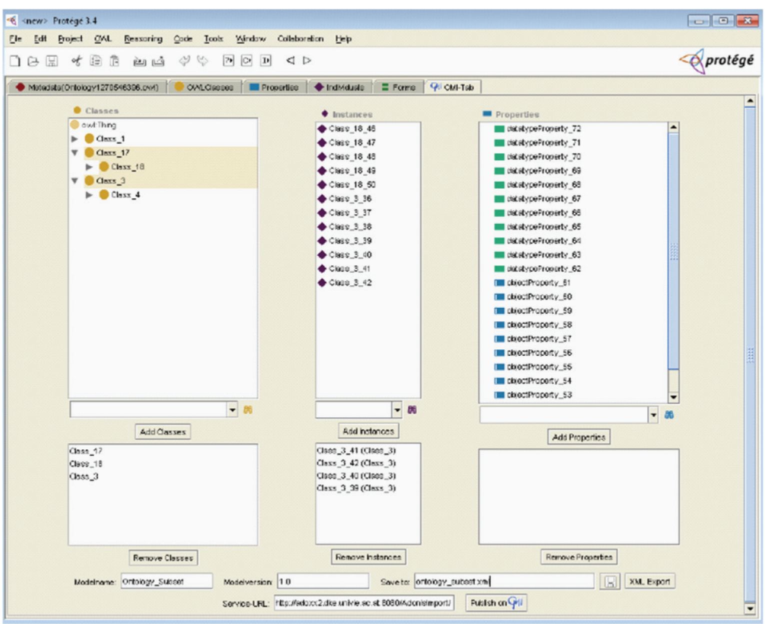
Protégé client can transfer ontologies to the load balancer service and store them as models. In the current configuration this step requires that a meta model is available in the ADOxx®-WS that allows to store the ontology information as has been described in (Fill & Burzynski, 2009). To support the access to the ontology/model transformation service a Protégé plugin has been created that exports the ontology information in a generic XML format (see Figure 5). The Protégé client can either be made available as a standalone Java application, as a Java applet embedded on a website or as a Java web start application.

On the side of the models, a web modeler service and a corresponding Java applet (see Figure 4) together with a documentation service and a JSP-based user interface implement the concepts of the model manipulation and model representation service. The interfaces of the transformation service and the load balancer service are described by WSDL documents. The exchange of messages with the interfaces described by WSDL is based on SOAP.

Based on this architecture the user is able to create or re-use existing ontologies using the Protégé client and store the ontologies in a database. By switching to the developed Protégé plugin, ontologies or parts of ontologies can be made available in the modeling environment as models. These ontology models can then be used for conducting the annotation of other models. Thereby the annotations are currently being stored as inter-model references. All models can be manipulated using the modeler applet together with the web modeler service. For documentation purposes that do not require a write access to the models the documentation service can be used.

The architecture and the adapted tools will be made publicly available under an open access license in the course of the Semantic based Modeling Framework for Information Systems (SeMFIS) project of the Open Model Initiative[6]. Thereby, it is envisaged to receive feedback from the community and further develop the approach.

Figure 5. Screenshot of the Protégé plugin



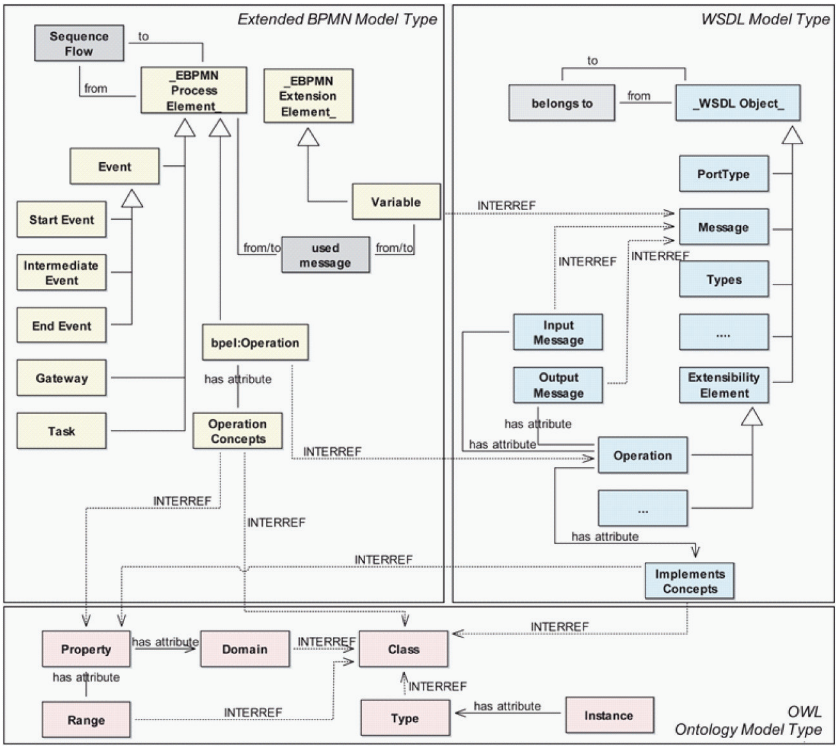
Sample Scenario for Using the Annotation Architecture

To illustrate the concrete usage of the described architecture we will use an example from the area of business process management. One particular application of semantically annotated business process models is to realize a dynamic discovery of services that can execute tasks specified in a business process. Through the semantic annotation these services need not be directly known but can be automatically suggested to a user based on their semantic characterizations including their operations, operation parameters and quality of service parameters. Thereby, the service parameters are semantically described using an ontology. A simple example could be the process task "send notification by e-mail". An ontology may contain concepts for "send", "e-mail", and "notification" that can be linked to this task in the process model. In the same way, one or more service operations may be linked to the same or semantically similar concepts of the ontology. These specifications will then permit an algorithm to search for services that can be used to execute the task in the business process by using the semantic relationships laid down in the ontology and suggest the services to the user. Additional annotations about the required service quality may even further narrow the search for appropriate services. Although the search for services may also be done manually, this approach may be of particular benefit when dealing with large numbers of services and processes that are developed by many different users. Then, the additional semantic information about the service requirements that is expressed through the semantic annotations may greatly ease the selection of appropriate services.

For the implementation of such a scenario the following steps are necessary: At first, a meta model for the definition of process models needs to be set up in the ADOxx® environment (Fill, 2004). For our purposes we selected the widely used business process modeling notation (BPMN) and extended it slightly for directly specifying linkages to service definitions in the web service definition language (WSDL). This includes the specification of the abstract model syntax and the corresponding notations. Next, an ontology meta model needs to be defined in ADOxx® for representing ontologies as models. In our example we chose the web ontology language (OWL) as one of the most widespread ontology languages. These two meta models can then be linked, thus specifying the elements of the business process to be annotated with ontology concepts. An excerpt of the integrated meta model is shown in Figure 6. Other standards that could be used for this integration include OWL-S or WSDL-S. However, they are more oriented towards the actual execution and do not focus on modeling. They are therefore more appropriate to be used as export formats and the actual execution on a workflow engine. After deploying the integrated meta model in the ADOxx® environment, a range of ADOxx® web services become active for accessing the meta models and offering functionalities for creating models.

In order to conduct the annotation, ontologies need to be made available in the modeling environment of ADOxx®. The linkage between the ontology management platform Protégé and the ADOxx® modeling environment is illustrated by the sequence diagram in Figure 7. It shows how ontologies can be exported using the plugin for Protégé, transformed via the ontology/model transformation service and then stored in one of the ADOxx® instances as selected by the load balancer service. It has to be noted that in the current conception of the architecture the notification on the successful storage of the ontology model in the ADOxx® environment is currently not reported back to the Protégé client, which is mainly due to the stateless conception of the architecture. With the models and ontologies both being available in the ADOxx® environment, one or more users can start to create the various models and assign the annotations. These are stored as references between the extended BPMN model and the OWL ontology model and one or more WSDL

Figure 6. Excerpt of a sample meta model for the semantic annotation of extended BPMN models

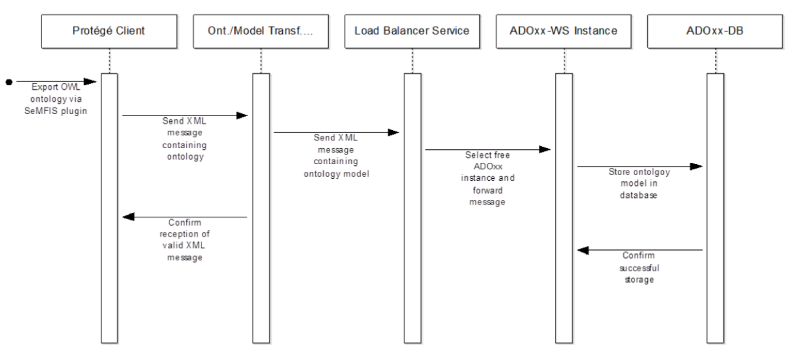


models and the OWL ontology model. By accessing the loadbalancer service of ADOxx® via its WSDL interface an algorithm can now access the models and their references. It can thus process the contained information and realize the above described service discovery.

Discussion

The above outlined architecture offers concepts for the flexible integration of different tools for handling of conceptual models and ontologies. As it is based on open SOA standards such as

Figure 7. Sequence of the service interactions for storing ontologies as models



XML, WSDL, and SOAP it can also be easily integrated with other platforms and operating systems.

Despite these advantages, there are also some open points that need to be taken into consideration when using it in practice. As the described approach is currently only a scientific research project it is not yet appropriate for full industrial usage. This concerns both the scalability of the approach as well as aspects such as backup and versioning mechanisms. As the architecture builds upon open standards it is required that all tools that shall be integrated support the chosen standards or offer at least options to extend the tools to do so. This may present a limitation to the applicability of the approach to other tools. For some interface definitions - such as the model exchange between different services that is based on XML - more detailed standards such as XMI could be used. However, this has to be evaluated based on the requirements of the involved communities and may be incorporated in a future version.

CONCLUSION AND OUTLOOK

We have described a generic service oriented architecture to support the semantic annotation of conceptual models. The architecture has been implemented using the ADOxx® platform and the Protégé tool. Due to the incorporation of service oriented aspects it is open to new developments and can be easily extended to support other ontology and modeling languages as well as other tools working on ontologies or models. Further work will include the extensive testing of the platform also in real-life scenarios. Future research work will be done in particular to investigate the dependencies between the different services in the framework and to take into consideration additional aspects such as model and ontology versioning and the propagation of changes between the services. Also the integration of reasoning and analysis services will be a future topic.

ACKNOWLEDGEMENT

We would like to thank Jennifer Vendetti from Stanford University for discussing the details of the Protégé architecture. Parts of the work on this paper have been funded by the Austrian Science Fund in the course of an Erwin-Schrödinger-Fellowship grant number J3028-N23.

REFERENCES

Abramowicz, W., Filipowska, A., Kaczmarek, M., & Kaczmarek, T. (2007). Semantically enhanced business process modelling notation. In M. Hepp, K. Hinkelmann, D. Karagiannis, R. Klein, & N. Stojanovic (Eds.), *Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007)*. CEUR Workshop Proceedings.

Atkinson, C., Gutheil, M., & Kiko, K. (2006). On the relationship of ontologies and models. In *Meta-modelling and ontologies (womm 2006)* (pp. 47–60). Gesellschaft fuer Informatik.

Borgida, A., Chaudhri, V., Giorgini, P., & Yu, E. (2009). *Conceptual modeling: Foundations and applications*. Berlin, Germany: Springer. doi:10.1007/978-3-642-02463-4.

Born, M., Hoffmann, J., Kaczmarek, T., Kowalkiewicz, M., Markovic, I., Scicluna, J., et al. (2008). Semantic annotation and composition of business processes with maestro. In *Proceedings of the 5th European Semantic Web Conference* (ESWC 2008). Springer.

Ehrig, M., Koschmider, A., & Oberweis, A. (2007). Measuring similarity between semantic business process models. In J. Roddick & A. Hinze (Eds.), *Proceedings of the fourth asia-pacific conference on conceptual modelling (apccm 2007)* (Vol. 67, pp. 71–80). ACM.

Erl, T. (2005). *Service-oriented architecture: Concepts, technology, and design*. Upper Saddle River, NJ: Prentice Hall PTR.

Fill, H.-G. (2004). UML statechart diagrams on the ADONIS metamodeling platform. *Electronic Notes in Theoretical Computer Science*, *127*(1), 27–36. doi:10.1016/j.entcs.2004.12.028.

Fill, H.-G. (2009). *Visualisation for semantic information systems*. Wiesbaden, Germany: Gabler. doi:10.1007/978-3-8349-9514-8.

Fill, H.-G., & Burzynski, P. (2009). Integrating ontology models and conceptual models using a meta modeling approach. In *Proceedings of the 11th International Protégé Conference*, Amsterdam, Netherlands.

Fill, H.-G., & Reischl, I. (2009). An approach for managing clinical trial applications using semantic information models. In S. Rinderle-Ma, S. Sadiq, & F. Leymann (Eds.), *Business process management workshops 2009* (pp. 581–592). Ulm, Germany: Springer.

Höfferer, P. (2007). Achieving business process model interoperability using metamodels and ontologies. In H. Oesterle, J. Schelp, & R. Winter (Eds.), *15th European Conference on Information Systems (ECIS2007)* (p. 1620-1631). St. Gallen, Switzerland: University of St. Gallen.

Karagiannis, D., Grossmann, W., & Höfferer, P. (2008, 0404-2010). Open model initiative -a feasibility study. Retrieved from http://cms.dke.univie.ac.at/uploads/media/OpenModelsFeasibilityStudySEPT2008.pdf

Karagiannis, D., & Kühn, H. (2002). Metamodelling platforms. In *Proceedings of the third international conference on e-commerce and web technologies* (p. 182). London, UK: Springer-Verlag. Retrieved November 15th, 2010, from http://www.dke.univie.ac.at/mmp/FullVersion_MMP_DexaECWeb2002.pdf

Kiryakov, A., Popov, B., Terziev, I., Manov, D., & Ognyanoff, D. (2004). Semantic annotation, indexing, and retrieval. *Web Semantics: Science. Services and Agents on the World Wide Web*, *2*(1), 49–79. doi:10.1016/j.websem.2004.07.005.

Lautenbacher, F., & Bauer, B. (2007). A survey on workflow annotation & composition approaches. In M. Hepp, K. Hinkelmann, D. Karagiannis, R. Klein, & N. Stojanovic (Eds.), *Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007)*.

Lautenbacher, F., Bauer, B., & Seitz, C. (2008). Semantic business process modeling -benefits and capability. In *Proceedings of the AAAI 2008 Stanford Spring Symposium - AI Meets Business Rules and Process Management (AIBR)*.

Musen, M. A. (1989). An editor for the conceptual models of interactive knowledge-acquisition tools. *International Journal of Man-Machine Studies*, 30.

Mylopoulos, J. (1992). Conceptual modeling and telos. In P. Loucopoulos, & R. Zicari (Eds.), *Conceptual modelling, databases and case: An integrated view of information systems development* (pp. 49–68). Wiley.

Obrst, L. (2003). Ontologies for semantically interoperable systems. In *Proceedings of the 12th international conference on information and knowledge management* (pp. 366-369). New Orleans, LA: ACM Press.

Singh, M. P., & Huhns, M. N. (2005). *Service-oriented computing: Semantics, processes, agents*. John Wiley & Sons, Ltd..

Thomas, O. (2007). Joint reference modeling: Collaboration support through version management. In *Proceedings of the 40th Annual Hawaii International Conference on System Sciences (HICSS'07)* (pp. 9a). Waikoloa, Big Island, HI: IEEE Computer Society.

ENDNOTES

1 WordNet® is a freely available online thesaurus that can also be accessed programmatically. See http://wordnetweb.princeton.edu.

2 See http://www.w3.org/standards/webofservices/ (accessed 04-04-2010)

3 Triple stores are specific types of databases for storing and retrieving ontology information in the form of subject-predicate-object constructs.

4 ADOxx®, ADONIS®, ADOscore®, ADOlog®, and ADOit® are registered trademarks of BOC AG. A free community edition of ADONIS® is available at http://www.adonis-community.com/

5 See http://protege.stanford.edu

6 For further information see http://www.openmodels.at/web/semfis

Hans-Georg Fill is an assistant professor at the University of Vienna. He holds a master's degree in international business administration and a PhD in business informatics from the University of Vienna. His research interests include semantic information systems, visualization and meta modeling and their application in business informatics. Most recently he received an Erwin-Schrödinger fellowship from the Austrian Science Fund to conduct a one-year research project at Stanford University. He is actively engaged in the scientific community and serves on several program committees and review boards in business informatics and information systems.

Daniela Schremser holds a bachelor's degree in business informatics from the Vienna University of Technology. She currently finishes her master thesis at the University of Vienna in which she develops a meta model based repository concept. Her research interests include semantic technologies, meta modeling and their application in business informatics. Furthermore, she is actively engaged in the Open Model Initiative.

Dimitris Karagiannis is a full professor for computer science and head of the Department of Knowledge and Business Engineering at the University of Vienna. His research focuses on meta modeling and its applications within the Open Model Initiative OMI, www.openmodels.at. OMI is a community that deals with the creation, maintenance, modification, distribution, and analysis of modeling methods and domain-specific models. Furthermore, he is engaged in several nationally and internationally funded research projects. In 1995 he founded BOC AG, which is today one of the leading consulting and software houses for business process and enterprise architecture modeling.