17th International Conference on the Applications of Computer
Science and Mathematics in Architecture and Civil Engineering
K. Gürlebeck and C. Könke (eds.)
Weimar, Germany, 12–14 July 2006

SEMANTIC SERVICE ENVIRONMENTS FOR INTEGRATING TEXT WITH MODEL-BASED INFORMATION IN AEC/FM

S.-E. Schapke, R.J. Scherer, P. Katranuschkov[*]

[*]*Institute for Construction Informatics, Technische Universität Dresden, Germany*
E-mail: Sven.Schapke@cib.bau..tu-dresden.de

Keywords: Text Integration, Text Technologies, Semantic Web Services, Ontology

Abstract. *In distributed project organisations and collaboration there is a need for integrating unstructured self-contained text information with structured project data. We consider this a process of text integration in which various text technologies can be used to externalise text content and consolidate it into structured information or flexibly interlink it with corresponding information bases. However, the effectiveness of text technologies and the potentials of text integration greatly vary with the type of documents, the project setup and the available background knowledge.*

The goal of our research is to establish text technologies within collaboration environments to allow for (a) flexibly combining appropriate text and data management technologies, (b) utilising available context information and (c) the sharing of text information in accordance to the most critical integration tasks. A particular focus is on Semantic Service Environments that leverage on Web service and Semantic Web technologies and adequately support the required systems integration and parallel processing of semi-structured and structured information.

The paper presents an architecture for text integration that extends Semantic Service Environments with two types of integration services. Backbone to the Information Resource Sharing and Integration Service (IRIS) is a shared environment ontology that consolidates information on the project context and the available model, text and general linguistic resources. It also allows for the configuration of Semantic Text Analysis and Annotation Services (STANs) to analyse the text documents as well as for capturing the discovered text information and sharing it through semantic notification and retrieval engines. A particular focus of the paper is the definition of the overall integration process configuring a complementary set of analyses and information sharing components.

1 INTRODUCTION

In distributed project organisations and collaboration there is a need for integrating independent and self-contained text information with centrally managed project data. Even with an increasing integration of design, engineering and controlling systems, a multitude of text documents such as contracts, expert's reports, and notifications will prevail in practice due to various social, legal and technical reasons. In our research we consider this a process of *text integration* in which the text content can be either consolidated into structured information itself or flexibly interlinked with corresponding information bases. Both the structuring as well as the interlinking is first of all hampered by the complexity of natural language text. Common ambiguities of terms and phrases, the use of semantically vague expressions as well as the vagueness of most document based communication is a foundation of effective human communication but a challenge for the automatic processing of text contents.

Various text technologies based on linguistic, classification, search and data mining methods can be used to externalise relevant text information. However, due to the heterogeneity of project documents in AEC/FM as well as the distinctive functionality and effectiveness of the text mining and information retrieval technologies, the potentials as well as the importance of text integration greatly varies with a documents' focus and style, the actual project setup and the availability of background knowledge. In fact, while for some documents a few metadata attributes may be sufficient to assign them to corresponding workflow tasks or building objects, the content of other documents may contain several aspects relevant to numerous users, processes and applications. Total information integration in one (logical or even physical) information base as pursued in CSCW or EAI appears neither necessary nor feasible.

We argue that text technologies can be most beneficially deployed in construction complementing existing data management technology for certain document types and business cases. Particularly, the different project applications provide up-to-date project context information to support the text analyses. Hence, the objective of this research is not to develop yet another text analysis or mining algorithm, but establish text technologies within collaborative environments to allow for combining the most appropriate text and data management technologies and share text information in accordance to critical integration tasks and available linguistic, industry and project knowledge. In this context objectives and requirements of text integration are:

- access to the text content of various project resources ranging from reports, specifications and emails to CAD drawings,
- context information on the setup and status of a project to specifically guide and support text analyses and knowledge discovery,
- access to various general, construction- and context-specific text technologies preferably via a common interface,
- retrieval models and a common business logic for sharing text information, (re-)integrating it with structured information and notifying responsible project participants, and finally
- integration technologies that allow for configuring and orchestrate the different integration process.

The required access to content and context information can first of all be provided by central project communication platforms and collaboration environments. We argue that, particularly project environments based on Web service and Semantic Web technology, called *Semantic Service Environments* in the following, are suitable for the required integration of distributed analysis systems and the parallel processing of semi-structured and structured information. The great amount of research in language engineering and text mining proliferated by the Web has lead to an increasing number of text standards and modular analysis systems that can be adapted to new environments and application domains.

The paper presents our approach to text integration that extends Semantic Service Environments with Web services for analysing project documents and sharing the externalised text information. At first, the essential features of Semantic Service Environments are described and their suitability for text integration is discussed. Then, the architecture of the suggested text integration services and their orchestration is presented. Following, the components the *Information Resource Sharing and Integration Service* (IRIS) and the *Semantic Text Analysis and Annotation Service* (STAN) are described in detail. Finally an ontology presented that provides for the configuration the document analyses and text sharing technologies in so called *Integration Scenario Definitions* (INSIDEs).

2 SEMANTIC SERVICE ENVIRONMENTS

Semantic Service Environments make use of the advantages of Web service and Semantic Web technologies to support process coordination and interoperability among distributed information systems. They have been recognised by several researchers to overcome shortcomings of the centralised project data management and project communication platforms commonly used in AEC/FM [12, 5]. In the following paragraphs the two technologies are shortly revisited and their support for text integration is discussed.

Web services technology provides for the web-based deployment of service-oriented architectures in which a system is composed of several services components [3, 1]. A Web service is an independent, self-describing, modular application that is located and invoked across the Web. With the publication of the profile and interface of a Web service, other applications (and services) can discover and interact with the service. A Web service environment is based on a set of standard protocol and interface specifications such as SOAP, WSDL and UDDI that provide basic means for Web service definition, registration, location and communication. The key advantage of Web services is on-the-fly software creation through the use of loosely coupled, reusable software components. Advantages over traditional middleware and broker architectures are the wide use of Web standards, its openness and support of heterogeneous platforms.

Web services are first of all recognised in e-business where the technology is implemented with several of today’s B2B and B2C applications. In AEC/FM, particularly central support systems e.g. for procurement or the management of errors and omission are deployed as Web services. However, most of these services implement distinctive interfaces and data structures and thus provide little support for data interoperability and project-wide information sharing.

The Semantic Web is an envisioned extension of the Web to define the meaning of information supporting the discovery, integration, and reuse of Web content [1, 6]. It is based on semantic annotations that capture additional metadata on the Web content in accordance to a well-defined semantics. The development of the Semantic Web comprises a hierarchically organised collection (so called Semantic Web Stack) of declarative languages to define and corresponding formalisms to process the annotations. Today, the most prominent standards used are the Resources Description Framework (RDF/ RDFS) that represents a basic data model of the Semantic Web [15] and the Web Ontology Language (OWL) that provides for defining ontologies based on description logic [13]. The main advantages of the Semantic Web are (a) the unique identification and common treatment of heterogeneous resources distributed throughout the Web, (b) the wide use of text- and web-based standards in particular XML and (c) the definition of metadata vocabularies by formal ontologies that provide for flexible processing and reasoning.

Semantic Web technology can not only be used to describe the content but also the service infrastructure itself. Particularly the two specifications OWL-S and the WSMO have been proposed for developing *Semantic Web Services* that allow for automatic discovery, composition and execution of Web services [8, 14, 1]. However, while the current concepts for Semantic Web services provide for coupling service functionalities for conventional business transactions they need to be extended to also support the collaborative use of heterogeneous information resources in construction. Hence, in a Semantic Service Environment the shared semantics is envisioned to allow for both: the identification and reuse of information resources as well as the orchestration of processing resources.

A central challenge for establishing Semantic Service Environments in AEC/FM is the development of a *shared environment ontology* that covers the different aspects of distributed collaborative working as well as the conceptual views of the different project participants. On the one hand such a development can bottom-up draw upon available semantic resources such as construction classifications and product data models which also eases a latter integration of corresponding information. On the other hand it should be kept as simple as possible to allow for easy adaption of existing applications and services.

To provide for a scalable infrastructure various layered ontologies have been proposed that ensure a minimum ontological commitment by their top layers. A most comprehensive ontology for knowledge management that embraces several AEC/FM models and standards has been developed in the e-Cognos project [9]. The design of an environment ontology that particularly describes the collaboration in virtual organisations is currently undertaken in the inteliGrid project [6]. The inteliGrid ontology framework distinguishes ontologies describing the project organisation, services and resources tied together by a business processes ontology. For each of the ontologies corresponding components are implemented that can be combined in a central Ontology Management Service to provide for a common interface to discover and retrieve meta-information on the distributed project information.

In our understanding such Semantic Services Environments constitute several features that are desirable if not a necessary prerequisite of a project infrastructure to adequately support the integration of text information. These are:

- The loose coupling of project services based on standard protocols provides for an easy integration of text analyses and integration services.
- The text-based specification of schemata as well as annotations provides for a uniform processing of text and model-based data information.
- The resource concept of the Semantic Web provides for a scalable approach to support work practices based on common documents, structured data as well as that it is expected to be extendable to model-based collaboration.

- The declaration of information based on Web technology and formal semantics provides for the interpretation and aggregation of distributed and incomplete information. This is believed to be a major advantage to provide for the collection and consolidation of the required project context information as well as for a better reuse of the isolated, partially sparse text information that can be automatically extracted from documents.
- The environment ontology explicitly describes not only the exchanged design and construction products but also the actors and processes involved in the project. This provides for a platform- and applications-independent specification of the processes for sharing text information as well as it supports direct access to context information normally used only within single applications.

3 ARCHITECTURE AND INTEGRATION SCENARIO

Figure 1 shows the architecture of a Semantic Service Environments for text integration. For a typical integration scenario, it illustrates the interactions among the common project and the developed text integration services.

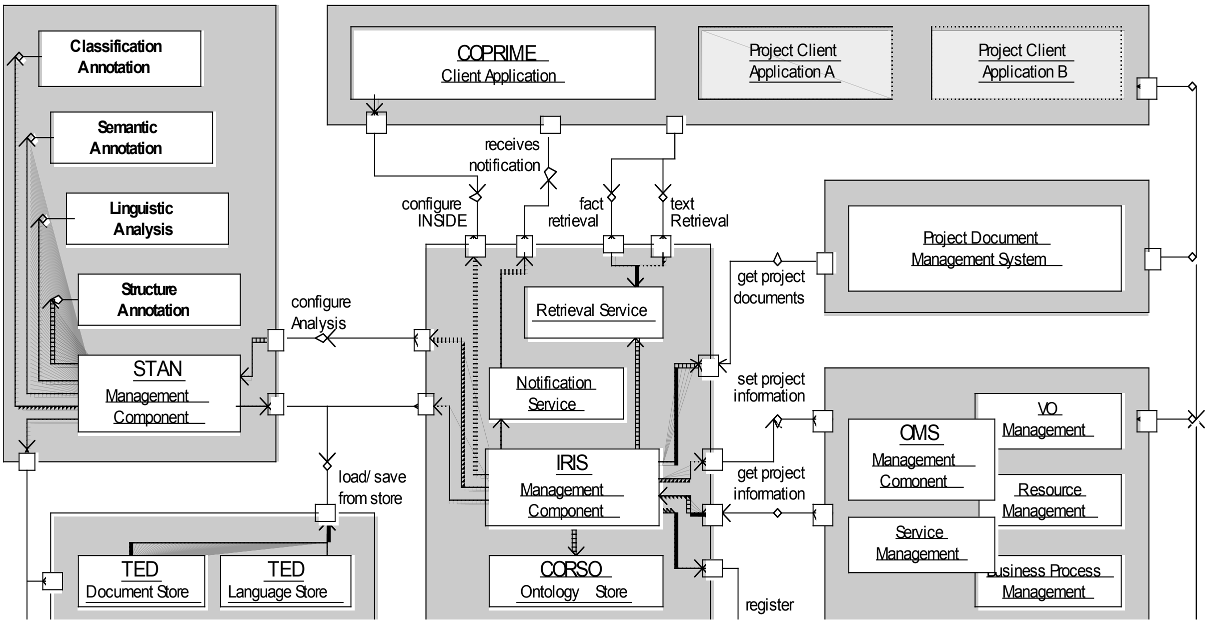

Figure 1: Architecture of Semantic Service Environment for Text Integration

Central to the approach is the *Information Resource Sharing and Integration Service (IRIS)* that controls the integration processes. It coordinates two types of subsidiary services, namely the *Text Document Store (TED)* that consolidates collections of t ext documents and linguistic resources as well as the *Semantic Text Analyses and Annotation Service (STAN)* that encompasses different text technologies for the documents' analysis. In the current version a single client, named *Construction Project Information Memory* (COPRIME) client, is used to configure the integration scenarios and display the results. However, for integrating text information on real-life projects it will be essential to implement functionalities for retrieving text information and receiving respective notifications with several project applications such as product data model as well as project and workflow management systems.

In the illustrated scenario, two data management services provide for the required project information. General project information is obtained from the *Ontology Management Serv-*

ice (OMS) that centrally manages meta-information on the operational project resources distributed throughout different application services. For a prototypical implementation of the infrastructure the inteliGrid Ontology Management Service is used. To provide for direct access to the required context information in the document analyses the collected information on project participants, processes, models, documents as well as the registered Web services is consolidated in the *Construction Information Resource Sharing Ontology (CORSO)*. The actual text documents are loaded from a common *Document Management System (DMS)* as used in most of today's project environments and communication platforms.

The information flow for the illustrated integration scenario is indicated by the grey numbers in figure 1. The starting point of the text integration is the deployment of an *Integration Scenario Definition (INSIDE)* that lays out the text integration process. Secondly, the type of text resources to be analysed are identified as specified in the scenario definition. The respective documents are than selected from the ontology base, loaded form the DMS and stored in TED Service. Thirdly, the text components of the STAN Service and the sequencing of the analyses are configured. For each analysis component, the scenario definition specifies the required parameters, training data, general linguistic and semantic resources as well as the resources that are still to be generated from the CORSO ontology, e.g. lists of project partners. Fourthly, the analyses results are captured in semantic annotations of text units. For each annotated text unit an ontology instance (text entity) can be generated and classified in the CORSO ontology providing for the (re-)integration of the recognised text information. Fifthly, the two IRIS components for sharing the text information are configured. The *Notification Service* can inform project services upon recognition of particular text content in a push mode. For retrieving text information in a pull mode, the *Retrieval Service* is envisioned to build up retrieval indexes from the text content as well as the annotations to support multiple text and fact retrieval models. Finally, the recognised text information can also be shared merging the ontologies of the IRIS and the OMS services and making it available in the course of operational project information sharing.

4 INFORMATION RESOURCE SHARING AND INTEGRATION SERVICE

The Information Resource Sharing and Integration Service (IRIS) comprises a central management component, two information sharing components and an underlying ontology store. However, the backbone of all components is the CORSO ontology. Consolidating information on the general project context as well as the services and text resource, it contains a great amount data that is already stored in Ontology Management Service. This parallel ontology has been designed because of the following:

- To support the manual configuration of integration scenarios the COROS ontology shall provide for easy browsing and a simple top-down concept hierarchy.
- To explicitly capture linguistic knowledge and define its interrelations with engineering concepts the ontology needs to be complemented with semiotic concepts.
- To provide for the automatic classification of text entities expressive assertions are required that are not necessarily provide by the discussed ontologies.
- Before the discovered text information is inserted into the environment ontology it needs to be consolidated, verified and complemented with information on its trustworthiness.

Currently the CORSO ontology is limited to a lightweight upper ontology that shall provide for substituting general concepts with respective domain models as required by the particular integration task and projects setup. A detailed description of the ontology is given in [10]. With a focus on the parallel handling of disparate information resources it mainly builds on three basic classes:

- *ConstructionProjectRealmEntity* (core) describing physical and abstract matters of real-life construction collaboration,
- *ProjectEntityRepresentation* (peer) describing the representations of core entities in different resources, e.g. lexical terms and phrases in natural language text,
- *EntityResource* (entry) specifying the resource that provided the definition of the core entity e.g. to distinguish between manually defined and automatically discovered facts.

The actual management of the ontology graph is performed by a management component based on the Jena framework that provides for the required querying and reasoning on the ontology model [7]. Basic functionalities are the insertion of INSIDEs, general linguistic knowledge and the discovered text entities. Hence, the fact base also includes the parameters for configuring and executing the STAN services and the import of text resources. For im- and export of project context information a bidirectional mapping among the CORSO and the inteliGrid ontology based on a simple mapping of top-level classes is aspired.

Two different types of information sharing components are suggested. A simple component is the Notification Service that sends the selected CORSO information to particular client services based on a predefined filter and listener to the CORSO store. On the contrary the research on retrieval models that can make use of both the text content as well as the available context information is still at its beginning. For the prototypical implementation of the infrastructure multiple term indexes are generated for differently unrecognised text tokens and selected annotated text units. Other semantic retrieval engines are discussed in [4, 1].

5 SEMANTIC TEXT ANALYSIS AND ANNOTATION SERVICE

The Semantic Text Analysis and Annotation Service (STAN) is composed of a central management and four abstract analysis components for structure annotation, linguistic annotation, semantic annotation and classification annotation (see figure 1). The analyses components reflect the classification of possible document analyses in the CORSO ontology. It shall provide for a uniform configuration of various analyses services as most existing text systems focus on selected analysis tasks and only comprise a few complementary linguistic and discovery components. In this context particular construction-specific STAN Services need to be developed to extract text information from e.g. standard forms and legal notifications.

The current implementation of the STAN Service is based on GATE, a component-based software architecture for language engineering that incorporates analyses components of several research projects [2]. In GATE an application is composed of three types of software components, namely (a) language resources such as lexicons and ontologies, (b) processing resources such as parsers and taggers as well as (c) visual resources that provide for the assembly and control of the analyses. XML-based profiles are used to configure the components at run-time so that different variations of the generic processing resource can be built. This supports the flexible exposure of both preconfigured as well as configurable analyses functions at the STAN service interface. The corresponding approach to using OWL-S process specifications to define the analyses functions and parameters of a STAN service and execute complementary analyses in composite processes is described in section 6 below.

GATE has already been the basis of the DoKMoSis System for which selected German as well as construction specific language and processing resources have been developed by the authors [11]. For the STAN Service the system is deployed as a Web service and complemented with analyses components for a given document corpus of correspondences among a construction manager and subcontractors as follows:

- The structure analysis focuses on the extraction of the email metadata and the segmentation of longer documents into chapters and paragraphs.
- The linguistic analysis builds on the tokeniser and sentence splitter of GATE and revised versions of the coreferencer and wrapper to the German part-of-speech tagger TreeTagger.
- The semantic annotation first of all comprises entity recognition based on general construction domain lexica and finite-state transducers e.g. for extracting measurements, scales, regulations, materials. However, central to our approach is the recognition with context-specifically gazetteers and transducers that are automatically build from the project information e.g. for extracting project actors, addresses, contracts, etc. After the analyses an ontology-based postprocessor is used to instantiate CORSO classes for the recognised text entities.
- A text classification module based on a classical vector space model is to provide for the identifying certain type of messages such as RFIs, progress reports, change orders, etc.

6 INTEGRATION SCENARIO DEFINITIONS

While linguistic, retrieval and mining methods can be rather complex they are usually sequenced in simple linear analysis pipes. One goal of the envisioned infrastructure is the remote orchestration of distribute analyses systems in the analysis processes to allow for the utilisation of specialised algorithms and lexical knowledge of external providers. Moreover, as important as the composition of the analysis is the definition of information sharing processes to support the effective reuse of the discovered text information throughout the project.

To coordinate the overall integration process Integration Scenario Definitions are proposed that coordinate the delivery of project resources, the identification and execution of text analysis as well as the information sharing with the targeted model-based systems. Corresponding to extract, transform, load processes (ETL) in Data Warehousing they specify a complementary set of filter, analysis and transformation components for a particular integration task such as the automatic routing of incoming mails. The idea is that generalised INSIDEs for typical integration tasks can be preconfigured explicitly and than manually customised in accordance to the prevailing text information and the available context knowledge on a particular construction project.

Several technologies exist for an orchestration of distributed Web services such as the standards OWL-S and WSMO [8, 14, 1] or the Triana environment (www.trianacode.org). As the basis of the scenario definitions OWL-S was chosen, that solely leverages on OWL and supports capability based discovery, the automatic composition, and the automatic invocation of Web services. OWL-S does not replace Web services standards but provides an additional layer for Web service discovery (OWL-S/UDDI mapping) and invocation (OWL-S/WSDL Grounding). On this layer the Service Profile describes a service for its discovery and the Service Model specifies the actual ways a client may interact with the service.

In the Service Model the functions of a service are represented by processes that are characterized by its inputs, preconditions, outputs, and possible results. An overall process model is described as a tree structure in which Composite Processes being the internal nodes and Atomic/Simple Processes being the leaves are distinguished. Atomic Processes correspond to the basic function of a Web service hiding the details of its implementation and WSDL operations. Simple Processes provide a mechanism to define abstraction, yet unknown and hierarchical processes. Composite Processes finally lay out how processes work together to compute a complex function. They comprise a control flow specifying the temporal relations between the executed sub-processes as well as the data flow specifying the data transferred among the processes. Figure 2 depicts an exemplary composite process that describes the structural analysis and tokenisation of emails.

```
<process:CompositeProcess rdf:ID="BasicEmailAnalysesProcess">
    <process:hasInput rdf:resource="#EmailCorpus">
    <process:hasOutput rdf:resource="#AnnotatedEmailCorpus">
    <process:composedOf>
       <process:Sequence rdf:ID="Seq1">
           <process:components rdf:parseType="Collection">
              <process:Perform rdf:ID="Step1">
                 <entry:StructureAnnotationProcess rdf:ID="EmailAnnotationProcess">
                 <process:hasInput rdf:resource="#EmailEncoding"/>
              </process:Perform>
              <process:Perform rdf:ID="Step2">
                 <entry:LinguisticAnnotationProcess rdf:ID="TokenAnnotationProcess">
                     <process:hasInput rdf:resource="#AbbreviationList"/>
                     <process:hasInput rdf:resource="#PostTokenisationRules"/>
                 </entry:LinguisticAnnotationProcess>
              </process:Perform>
           ...
</process:CompositeProcess>
```

Figure 2: Simple Composite Process for the tokenisation of emails

Based on OWL-S an INSIDE ontology was developed that allows for defining (a) the name and type of integration scenarios, (b) the text documents to be considered, (c) the configuration and the sequencing of the analysis components, (d) the configuration and the process of the information sharing components as well as the (e) the trigger or temporal sequence for its execution. The INSIDE instances are stored in the CORSO store and used to coordination the corresponding data collection, text discovery and information sharing processes.

7 CONCLUSION

The effective deployment of text technologies in construction requires the flexible combination of various analysis systems and context information for document types and business cases. A Web service infrastructure is suggested that allows for the automatic coordination of overall integration scenarios configuring and executing the required documents analyses as well as the sharing of the discovered text information. Backbone to approach is a formal environment ontology that provides the basic means for the consolidating the available project information, capturing and classifying the discovered and partly incomplete text information as well as for using different semantic retrieval models.

REFERENCES

[1] H. P. Alesso, C. F. Smith, *Developing Semantic Web Services*, Peters, 2005.

[2] H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*. Philadelphia, USA, July 2002.

[3] W. Dostal, M. Jeckle, I. Melzer, B. Zengler, *Service-orientierte Architekturen mit Web Services - Konzepte, Standards, Praxis*. Elsevier, 2005

[4] L. Ding. T. Finin, A. Joshi, R. Pan, R.S. Cost, Y. Peng, P. Reddivari, V. Doshi, J. Sachs, Swoggle: A Search and Meatadata Engine for the Semantic Web. *Proceedings of the 13th Conference on Information and Knowledge Management (CIKM04)*, Washington, USA, November 2004.

[5] A. Gehre, P. Katraniuschkov, V. Stankovski, R. J. Scherer, Towards Semantic Interoperability in Virtual Organisations. *Proceedings of the 22nd CIB-W78 Conference on Information Technology in Construction*, Dresden, Germany, July 2005.

[6] A. Gehre, P. Katranuschkov, J. Wix, J. Beetz , D31: Ontology Specification. *inteliGrid project deliverable*, April 2006.

[7] Jena, Jena - A Semantic Web Framework for Java. *Developer Project Websites at http://jena.sourceforge.net/*, visited May 2006.

[8] R. Lara, D. Roman, A Polleres, D. Fensel: A Conceptual Comparison of WSMO and OWL-S. *Proceedings of ECOWS European Conference on Web Services*, Erfurt, Germany, September 2004.

[9] C. Lima, C. Ferreira da Silva, P. Sousa, J. P. Pimentão, C. Le-Duc, Interoperability among Semantic Resources in Construction: Is it Feasible?. *Proceedings of the 22nd CIB-W78 Conference on Information Technology in Construction*, Dresden, Germany, July 2005.

[10] S.-E. Schapke, R. J Scherer, Text Integration based on a Construction Information Resource Sharing Ontology. To be published in: *Proceedings of the European Conference on Product and Process Modeling*, Valencia, Spain, September 2006.

[11] S.-E. Schapke, R. J. Scherer, Interlinking Unstructured Text Information with Model-Based Project Data: An Approach to Product Model Based Information Mining. *Proceedings of the European Conference on Product and Process Modeling (ECPPM)*, Istanbul, Turkey, September 2004.

[12] A. Zeeshan, A. Chimay, R. Darshan, C. Patricia, B. Dino, Semantic Web Based Services for Intelligent Mobile Construction Collaboration. *ITcon Vol. 9, Special Issue Mobile Computing in Construction*, 2004

[13] W3C, OWL Web Ontology Language Overview, *W3C Recommendation at http://www.w3.org/TR/owl-features/*, February 2004.

[14] W3C, OWL-S: Semantic Markup for Web Services, *W3C Member Submission at http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/,* November 2004

[15] W3C, RDF Vocabulary Description Language 1.0: RDF Schema. *W3C Recommendation at http://www.w3.org/TR/rdf-schema/*, February 2004.