USING UDDI FOR PUBLISHING METADATA OF THE SEMANTIC WEB

Anton Naumenko, Sergiy Nikitin, Vagan Terziyan, Jari Veijalainen[*]
Industrial Ontologies Group, Department of Mathematical Information Technology, University of Jyvaskyla, FINLAND, e-mail: annaumen@cc.jyu.fi

[*] *Information Technology Research Institute, Faculty of Information Technology, University of Jyvaskyla, FINLAND, e-mail: veijalai@cc.jyu.fi*

Abstract: Although UDDI does not provide support for semantic search, retrieval and storage, it is already accepted as an industrial standard and a huge number of services already store their service specifications in UDDI. Objective of this paper is to analyze possibilities and ways to use UDDI registry to allow utilization of meta-data encoded according to Semantic Web standards for semantic-based description, discovery and integration of web resources in the context of needs of two research projects: "Adaptive Services Grid" and "SmartResource". We present an approach of mapping RDFS upper concepts to UDDI data model using tModel structure, which makes possible to store semantically annotated resources internally in UDDI. We consider UDDI as an enabling specification for creation of a semantic registry for not only services, but also for web resources in general.

Keywords: Web-Services, UDDI, Semantic Web

1. INTRODUCTION

Objective of the paper is to analyze possibilities and ways to use UDDI [UDDI] registry to allow utilization of meta-data, encoded according to Semantic Web [SemanticWeb] standards, for semantic-based description, discovery and integration of web resources in a context of needs of two research projects: "Adaptive Services Grid" (ASG) [ASG] and "SmartResource" [SmartResource], [Kaikova2004].

According to a definition by Moreau *et al* [Moreau2005] Semantic Discovery is the process of discovering services capable of meaningful interactions, even though the languages or structures, with which they are

described, may be different. In the paper authors evaluate existing approaches, basically, UDDI with keywords-based search, describe a solution to extend service descriptions using RDF [RDF] and changes to UDDI APIs needed to support a semantic search.

A description of entities using Semantic Web standards is called a semantic annotation or simply an annotation. The annotation of an entity is a prerequisite to allow semantic discovering and integration. In the context of UDDI, an entity of the semantic annotation is usually a Web Service and more rarely businesses, business services and technical information that is a target of a binding. We go beyond trying to consider UDDI as a basis to create a semantics-enabled registry of resources from point of view of a Semantic Web domain. Moreover, we consider each resource entity (not just a web service) as a subject of the semantic annotation, registering, discovering, composition, enactment, integration, etc.

Different attempts to bring semantics to UDDI were faced in a number of papers. They consider mainly the process of publishing semantic information to UDDI registry with or without changes to existing UDDI APIs and data model. Additionally, they focus on the process of a semantic search based on an internal enhanced matchmaker with changes to UDDI APIs or an external matchmaking engines through creating a proxy API above UDDI, matchmaker and ontology.

UDDI+ server [Pokraev2003] is a good example of a solution when UDDI is used unchanged, but inside architecture of the server, which introduces additional elements like a matchmaker, an ontology repository and a proxy API to invoke UDDI APIs. Such the solution requires mapping a semantic language, in this case DAML-S [DAML-S], to UDDI publish message while keeping standard UDDI Publish and Inquiry interface.

Nowadays, some research efforts are focusing on experiments with commercial UDDI registries [Kawamura2003], [Kawamura2004], [Paolucci1], [Paolucci2] trying to provide a semantic search based on an externally created and operated matchmaker. Web Service Semantic Profile (WSSP) serves as the semantic annotation of a service and extends WSDL [WSDL] description of the service using RDF, RDFS [RDFS], DAML+OIL [DAML] or OWL [OWL], RDF-RuleML [RuleML]. Semantic data are stored outside of UDDI while keeping a link from corresponding tModel of a Web Service registered with UDDI to its WSSP.

Srinivasan *et al* [Srininvasan2004] provides a research close to target of this paper and describes a mapping of an OWL-S profile to the UDDI data model for a matchmaker architecture based on the Paolucci's results [Paolucci1]. The difference from our approach is that the OWL-S to UDDI mapping is done on a conceptual level while in this paper we try to map an underlying RDF model to a structure of tModel.

The remaining content is organized as follows. Chapter 2 summarizes needs of the ASG project to create a service registry to store semantic descriptions of Web- and Grid Services in addition to other semantically encoded information like rules, facts, domain knowledge, etc. Chapter 3 provides a description of a business use case of the SmartResource project and needs of a registry for semantically annotated resources. Chapter 4 shortly presents architecture and information model of UDDI. Chapter 5 describes an approach to bring semantics to UDDI by encoding RDFS upper concepts using the data model of UDDI tModel. Chapter 6 concludes the results of the analysis performed.

2. SEMANTIC WEB AND RESOURCE DESCRIPTION FRAMEWORK

The Semantic Web is an idea of World Wide Web inventor Tim Berners-Lee that the Web as a whole can be made more intelligent and perhaps even intuitive about how to serve user's needs. Nowadays Semantic Web Activity has produced several standards for a specification of arbitrary domain knowledge with a rich semantic expressiveness.

Semantic Web is expected to become a next-generation of the Web assuming that besides an existing content there will be a conceptual layer of machine-understandable metadata, making the content available for processing by intelligent software. This allows automatic resource integration and provides interoperability between heterogeneous systems. The next generation of intelligent applications will be capable to make use of such metadata to perform resource discovery and integration based on its semantics. Semantic Web aims at developing a global environment on top of Web with interoperable heterogeneous organizations, applications, agents, web services, data repositories, humans, and so on. On the technology side, Web-oriented languages and technologies are being developed (e.g. RDF [RDF], OWL [OWL], OWL-S [OWLS], WSMO [WSMO], etc.), and the success of the Semantic Web will depend on a widespread industrial adoption of these technologies. A trend within worldwide activities related to Semantic Web definitely shows that the technology has emerging grows of an interest both academic and industry during a relatively small time interval. The growing interest to the Semantic Web, as a research and educational domain, from the academy is evident. New scientific results and interesting challenges in the area appear rapidly. International networks cover topics related to intersections of various former scientific domains with Semantic Web technology and discover new challenging opportunities. Basic standards have been announced and the amount of pilot tools and

applications around these standards is exponentially increasing. In spite of the growing hype around Semantic Web and appropriate standards, industry developed and is continuously developing own standards for interoperability and integration.

The Resource Description Framework (RDF) is a framework for representing information in the Web [RDF]. It is intended for integration of a variety of applications using XML [XML] for syntax and URIs for naming [SemanticWeb]. The RDF is a structure for describing and interchanging metadata on the Web [Powers2003]. The RDF is expressive and flexible technology to describe arbitrary domains and thus it is widely applicable. The World Wide Web Consortium (W3C) has been designing RDF as a basis technology to support Semantic Web Activity and it gives the following statement to describe the RDF: *The RDF is a language designed to support the Semantic Web, in much the same way that HTML is the language that helped initiate the original Web*.

The RDF is a framework for supporting resource description, or metadata (data about data), for the Web. RDF provides common structures that can be used for interoperable XML data exchange [SemanticWeb]. The RDF gives tools to developers to encode meaning by expressing concepts of problem domain and relations between them using RDF statements and connecting these statements to a semantic network. RDF, like XML and relational databases, follows object-based domain decomposition for data representation, but remains more generic and more expressive. There are also variety of software tools to work with RDF including tools for creating RDF, for creating vocabulary for RDF called Schema (RDFS), for querying RDF, for making inference based on an RDF defined semantic network, etc.

RDF brings to XML technology the same functionality as relational algebra to commercial database systems. RDF defines classes of problem domain concepts and their properties to create a vocabulary of the domain in the same way like a creation of tables and relationships between tables defines a schema of a database. XML can encode contents of a relational database and can encode the contents of an RDF-based model – but XML is not a replacement because XML is nothing more than syntax. A metadata vocabulary is needed to be able to use XML to record business domain information in such a way that any business can be documented, and RDF provides this capability [Powers2003].

3. ASG APPROACH TO DESCRIPTION, DISCOVERY AND INTEGRATION OF SERVICES

The main objective of the ASG-project is to develop an open generic software platform for adaptive services discovery, creation, composition and enactment. The ASG-platform is divided into several components having separate roles and interacting with each other by means of interfaces. The component structure is presented in Figure 1.

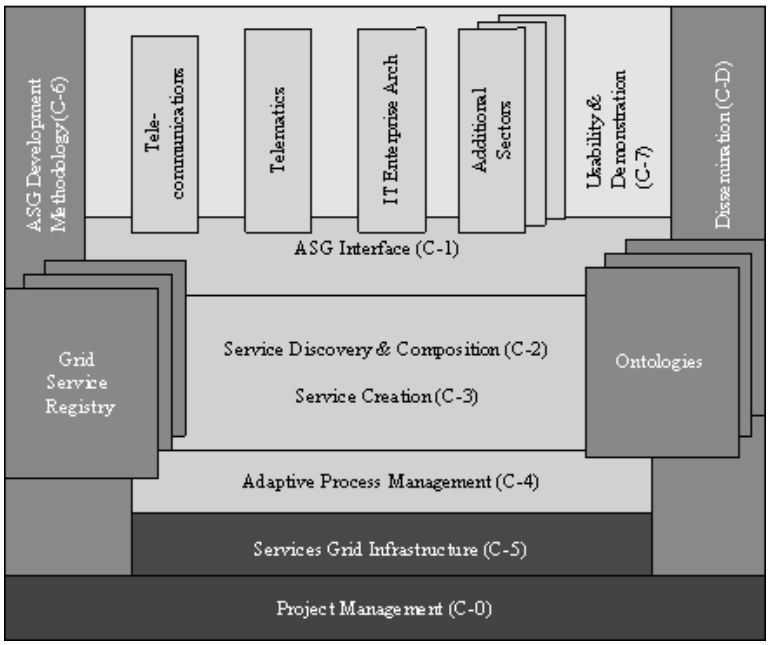

Figure 1. Component Structure (adopted from [ASG])

The key element of the ASG-platform is a persistent storage for platform data, so-called Registry. To understand a role of Registry, first let us consider business use cases.

In the following use cases a role of the Registry is quite straightforward. Registry represents the centralized platform storage containing various data about existing services and ontology specifications for domains involved in the service execution. A scenario used in the following use cases is a Traveling Tourist Scenario. It has following assumptions:

- User goal
  - Plan travel route from a departure location to a specific destination
- User specifies travel parameters:

- Stopovers (e.g. city), preferred travel means, max. cost, travel duration, points of interests, etc.,

- Possibly needed services:
  - Find nearby Point of Interests (POI)
  - Find nearby Hotels
  - Receive several alternative Travel Routes for including stay (Hotels) and sightseeing (POI)

To achieve the desired user's goal, the ASG-platform provides a workflow which defines the sequences and an order of the sequence peers invocation. The inputs and outputs of services are adjusted using grounding to a common ASG-ontology.

The use-cases of the ASG-project constitute quite disperse requirements to Registry. Registry must store atomic service specifications, composed of a service name and a specification of its inputs, outputs, pre- and post-conditions. As far as all parameters refer to the common platform and the domain ontology, this ontology must be stored too. Registry should also store specifications of composed services (represented as a workflow of other service invocations). Another important functional requirement to the Registry is an existence of management interface to all stored data. This interface should provide a set of methods to add, modify, delete and update Registry data. Particularly, when a new service is registered in the ASG-Registry, Registry stores a new service specification. The platform ontology is also a non-static component. It may evolve over time and thus causes a need to modify and extend ontology data. Figure 2 shows an adaptive process management and the role of Registry in it.

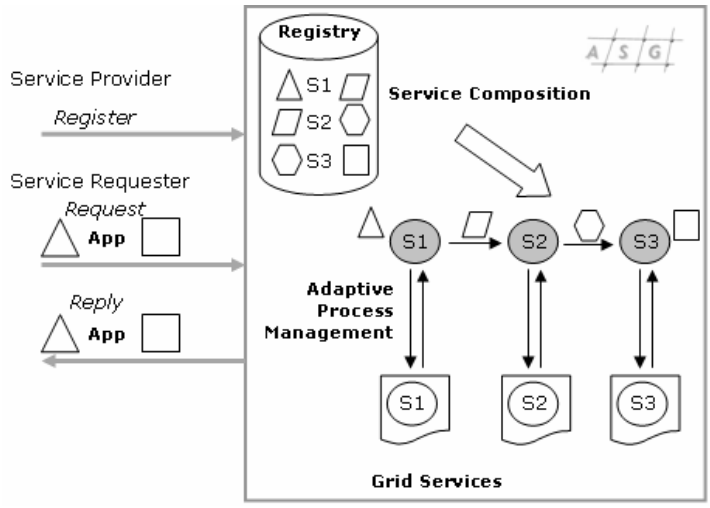

Figure 2. Role of the Registry in composed service invocation in ASG (adopted from [ASG])

From the point of view of non-functional requirements Registry should be a highly reliable and fault-tolerant component. This requirement is crucial, because the ASG-platform architecture implies conceptually centralized and technically distributed solution. Another challenge is scalability. As far as the role of Registry is to store everything about platform data, it becomes necessary to provide a solution capable of managing big data arrays.

ASG-platform requires from Registry to be capable of storing Service Specifications which in turn are done according to ASG-Ontology and refer to domain ontologies of corresponding domains. ASG-Ontology plays a role of a meta-model for Service Specifications. Here further we will refer to a work done by WSMO Working Group [WSMO] towards reusing UDDI as a persistent storage of WSMO-Registry [WSMOReg]. ASG-Ontology is a result of elaboration of WSMO and mappings between them are needed to support specifics for ASG concepts. We consider them both as initiatives facing the same problems of storing (representing) semantically rich data about Web-Services in a Registry-oriented way.

According to the WSMO approach, the data model of UDDI is not extended but necessary WSMO properties are mapped into existing data slots in UDDI (e.g. BusinessEntity) or in customized slots (Identifier bags).

Figure 3 shows the relationships needed to map a WSMO service to a UDDI model. According to proposed approach, existent properties of a UDDI model are reused and WSMO-specific properties are added.

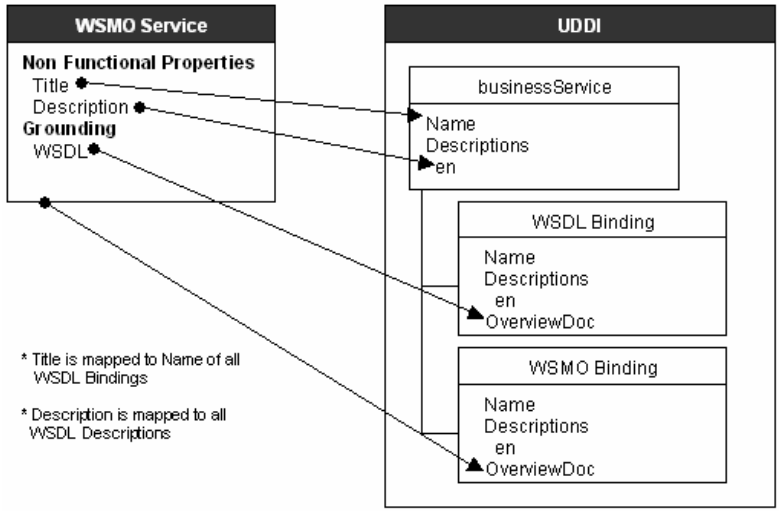

Figure 3. Mapping WSMO Service to UDDI (adopted from [WSMOReg])

4. NEEDS OF SMARTRESOURCE PLATFORM FOR REGISTRY, DISCOVERY AND INTEGRATION

Main source of functional and non-functional requirements for the Global Understanding Environment (GUN) platform is a set of business areas and use cases of the SmartResource project. In addition to existing business use cases from industry, where utilization of the SmartResource platform can be reasonable, the approach of SmartResource introduces new business opportunities and business models of an operation based on an open architecture of the SmartResource platform thanks to open standards of W3C and FIPA, which are the grounds of the platform design.

GUN [Kaykova2005] is a concept used to name a Web-based resource "welfare" environment, which provides a global system for automated "care" over (industrial) Web-resources with the help of heterogeneous, proactive, intelligent and interoperable Web-services. The main players in GUN are the following resources: service consumers (or components of service consumers), service providers (or components of service providers), decision-makers (or components of decision makers). All these resources can be artificial (tangible or intangible) or natural (human or other). It is supposed that the "service consumers" will be able: (a) to proactively monitor own state over time and changing context; (b) to discover appropriate "decision makers" and order from them remote diagnostics of own condition, and then the "decision makers" will automatically decide, which maintenance ("treatment") services are applied to that condition; (c) to discover appropriate "service providers" and order from them the required maintenance. Main layers of the GUN architecture are shown in Figure 4.

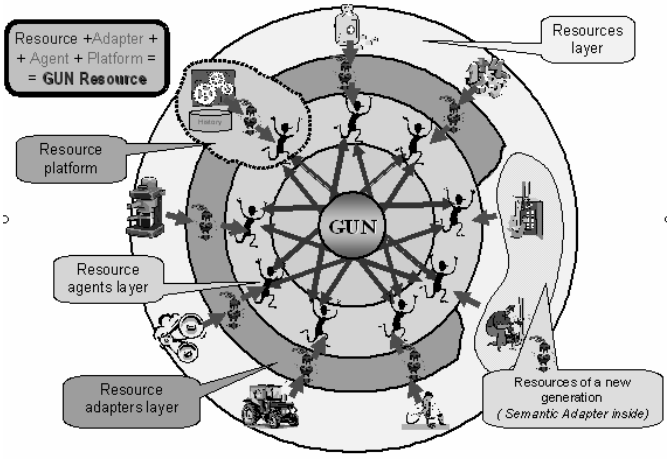

Figure 4. Layers of the GUN architecture

Industrial resources (e.g. devices, experts, software components, etc.) can be linked to the Semantic Web-based environment via adapters (or interfaces), which include (if necessary) sensors with digital output, data structuring (e.g. XML) and semantic adapter components (XML to Semantic Web). Agents are assumed to be assigned to each resource and are able to monitor semantically rich data about states of the resource coming from the adapter, decide if deeper diagnostics of the state is needed, discover other agents in the environment, which represent "decision makers" and exchange information (agent-to-agent communication with semantically enriched content language) to get diagnoses and decide if a maintenance is needed. It is assumed that "decision making" Web-services will be implemented based on various machine learning algorithms and will be able to learn based on samples of data taken from various "service consumers" and labeled by experts. Use of agent technologies within the GUN framework allows mobility of service components between various platforms, decentralized service discovery, utilization of FIPA communication protocols, and MAS-like integration/composition of services.

Condition monitoring of industrial devices is a target domain of the SmartResource project. Research prototype of the GUN environment in this project implements a use case of knowledge transfer from a diagnostic expert to a Web Service with a machine learning algorithm to substitute an expensive human resource by a diagnostic Web Service in the process of condition monitoring. Figure 5 illustrates the use case of a knowledge transfer.

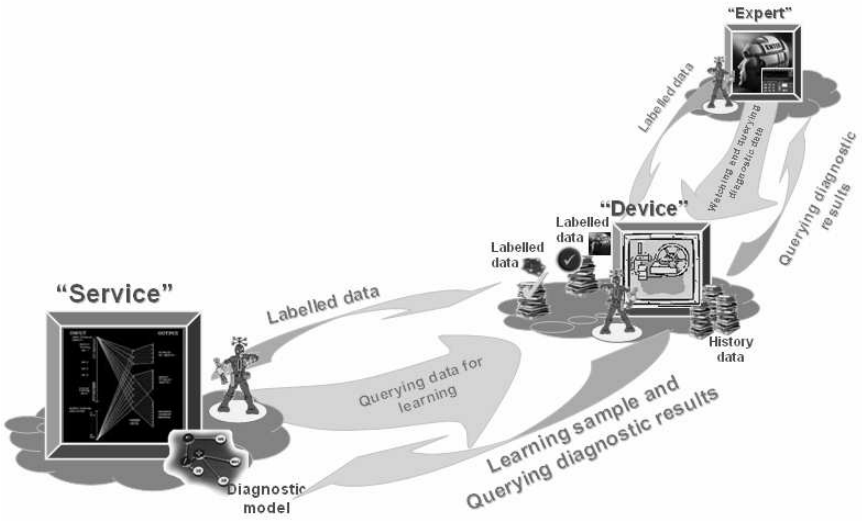

Figure 5. Knowledge transfer use case

However, the condition monitoring use case exists in different domains varying by an object of monitoring. To enable the use cases described above, one of the crucial research and development issues in the SmartResource project is to provide efficient mechanisms of description, discovery and integration of proactive resources.

Semantic Web provides standards for semantic description of resources in the Web that facilitates their discovery and integration. Such kind of a description for resources is called an annotation. RDF, RDFS and OWL cover aspects of a conceptual solution for a meta-data description in a form of ontology and description of a resource as an instance of certain class of resources using facets of this class in a typical case. The SmartResource project treats the entity description as a semantic annotation using Semantic Web standards.

Although an architectural and operational consideration of the process of a resource registration is out of scope of the Semantic Web standards, it is the most important issue for enabling automated discovery and integration. Thanks to the agent-driven platform of GUN, existing tools of multi-agent systems can solve partially tasks of registration, discovery and integration.

Despite of existing tools for multi-agent systems, semantic description, discovery and integration of resources is still an open question. We think that UDDI can be adapted to provide a functionality of a registry or a directory of semantically annotated resources that are SmartResources in the GUN platform.

5. A BRIEF DESCRIPTION OF A UNIVERSAL DISCOVERY OF DESCRIPTIONS AND THEIR INTEGRATION

It is attractive to use existing solutions for registering and discovering instead of implementing something else from scratch. Universal Description Discovery & Integration (UDDI) standard is the first one from the possible candidates, because it is widely spread and supported nowadays by big companies in their commercial and open source Registries.

We think that it is possible to use UDDI 3.0 [UDDISpec] for registration of semantically annotated entities without changes in the specification, while reasoning and other manipulations with the semantics would require changes to the specification of UDDI APIs. Focus of this paper is to provide a solution for mapping from concepts of Semantic Web standards to tModel concepts of UDDI. Possibility for an implementation of an advanced functionality for Registry based on Semantic data without changes to UDDI specification of APIs is also an issue to analyze.

UDDI by definition is a specification of services to provide publishing and discovery of "businesses, organizations and other Web Service providers", their Web Services and technical interfaces to enact those services [UDDIspec].

UDDI specification defines UDDI data model as a format for storing target entities of descriptions. Figure 6 illustrates the UDDI information model. Chapter 6 contains more details about the concept of tModel.

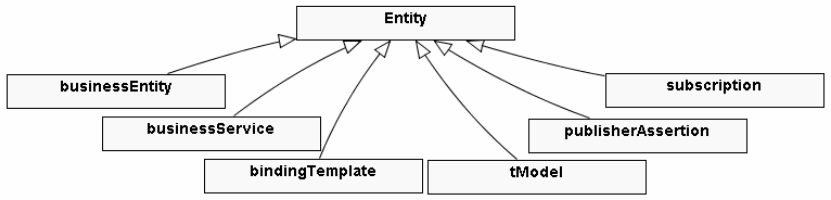

Figure 6. UDDI data model

Figure 7 shows sets of UDDI API defined in the standard.

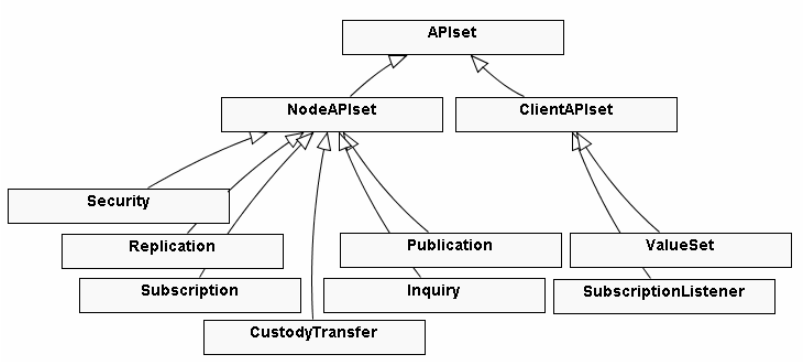

Figure 7. UDDI API sets

Figure 8 summaries the basic architecture of UDDI that allows a UDDI node to be an XML Web Service. The flexibility is achieved because UDDI does not restrict the technologies of the services, about which it stores information or the ways in which that information is decorated with metadata.

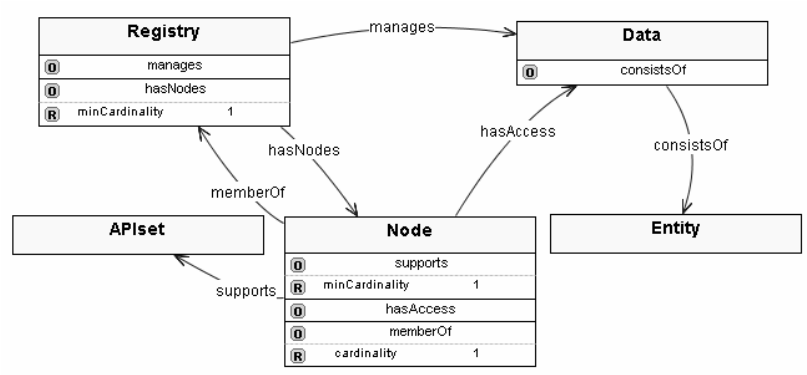

Figure 8. UDDI basic architecture

6. MAPPING OF ONTOLOGY CONCEPTS TO A UDDI DATA MODEL

The main advantage of using UDDI as a basis for an Ontology-based Registry is that a lot of mechanisms needed in Registry (like access rights, an administration, interfaces) are already defined, specified and implemented. Although UDDI does not provide support for semantic search, retrieval and storage, it is already accepted as an industrial standard and a huge number of services already store their service specifications in UDDI.

UDDI model contains a tModel element as a building block for storing different kinds of concepts and relations between them. tModel is a reusable concept, such as a Web service type, a protocol used by Web services, or a category system. The structure of tModel element is presented in Figure 10.

A *TModel* element must contain a name and a description and may contain *tModelKey* as a unique identifier. *tModel* may also have *identifierBag* and *categoryBag* elements. These elements are crucial as building parts of a structure of the ontology storage.

Structure of *identifierBag* is presented in Figure 11.

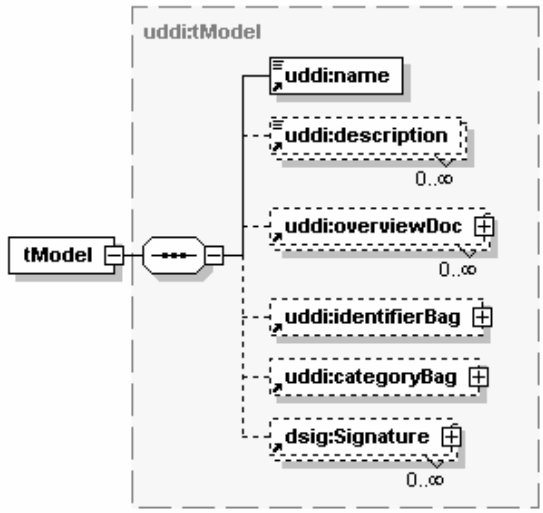

Figure 10. A tModel structure presented using an XML-Schema syntax

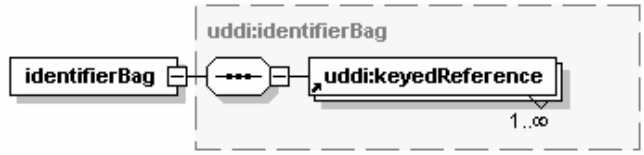

Figure 11. An *identifierBag* element

Thus *identifierBag* may contain 1 or more *keyedReference*s.

The *categoryBag* element has a little bit more sophisticated structure. It allows referring structures to be categorized according to published categorization systems. Figure 12 depicts the *categoryBag's* structure.

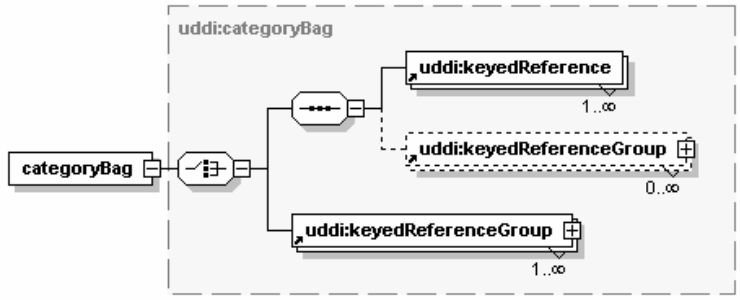

Figure 12. A categoryBag element

categoryBag may also contain directly one or more *keyedReference*s, but also allows a *keyedReferenceGroup* element, which in turn incorporates *keyedReference*s and must contain a *tModelKey* attribute that specifies the structure and meaning of *keyedReferences* contained in the *keyedReferenceGroup* (see Figure 13).

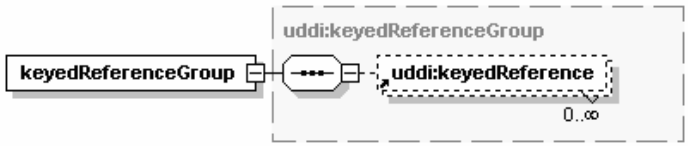

Figure 13. A keyedReference element

A *keyedReference* element, when included in *identifierBag*, represents an identifier of a specific identifier system. The *keyedReference* consists of the three attributes: *tModelKey*, *keyName* and *keyValue*. The required *tModelKey* refers to *tModel* that represents the system of identification, and the required *keyValue* contains the actual identifier within this system. The optional *keyName* may be used to provide a descriptive name for the identifier (see Figure 14).

```
<identifierBag>
    <keyedReference
        tModelKey="uddi:someidentifier"
        keyName="some descriptive name"
        keyValue="some value" />
</identifierBag>
```

Figure 14. Example of *keyedReference*

How this structure can be used to represent an ontology? Below we will consider a couple of examples of its application.

Example 1. *Ontology description language as a generic concept*.

In this approach we introduce a generic concept, e.g. “RDF Schema” and refer to it as to a set of concepts contained in RDF-Schema (Figure 15).

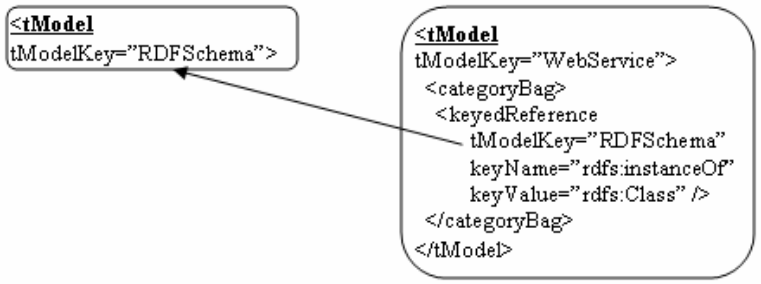

Figure 15. Introducing RDF-Schema as a tModel

After a quick view it looks reasonable to reuse a *keyName* element for defining the relationship between class *WebService* and the RDF-Schema concept *rdfs:Class*, referred by the *keyValue* element. However, when we want to create a subclass of a WebService class, say, *SemanticWebService*, then how to decide to which categorization scheme to refer? If we define tModel in a similar way (see Figure 16), we make a conceptual mistake, because *keyValue* of *keyedReference* has to point to a concept belonging to the set of concepts, defined by the referred *tModel*. In other words, *SemanticWebService* does not belong to the element set of "RDFSchema".

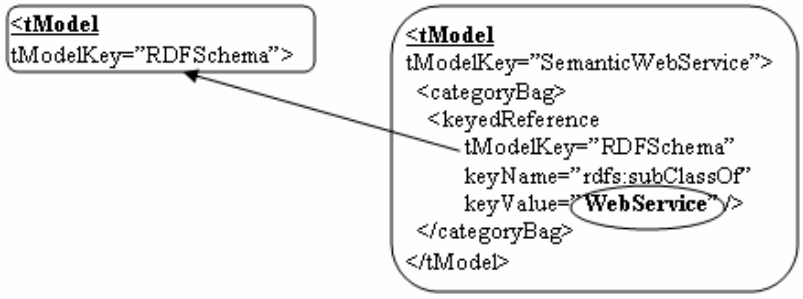

Figure 16. Wrong reference to the RDFSchema tModel

In this example we comply with the UDDI syntax, but contradict the semantics of UDDI concepts. It means that data stored according to this way will not be reusable by standard UDDI searching facilities.

Example 2. *Introducing concepts explicitly.*

In this case we reuse UDDI structures to represent classical relationship "*subject-predicate-object*" and build the ontology model on top of it. We map "*subject-predicate-object*" construction to the UDDI construction "*tModelKey-tModelKey-keyValue*" (see Figure 17).

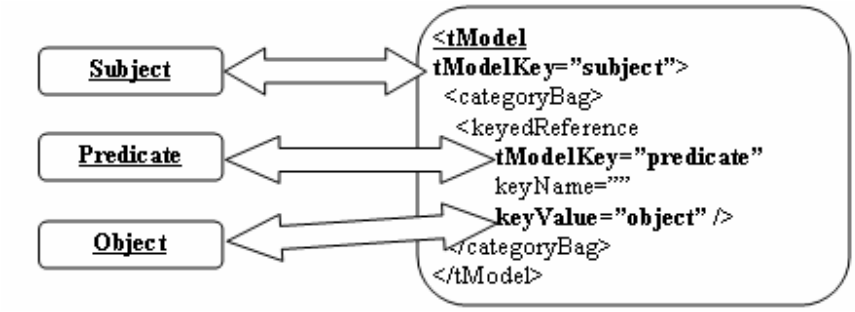

Figure 17. Concepts mapping

Let us continue with an RDF-Schema example. First of all we need to define RDF-Schema concepts as UDDI *tModels* (see Figure 18).

Figure 18. Definition of RDF-Schema concepts

As far as RDF-Schema is quite a big document, all concept mappings will not fit in the size of this paper, so we define only the crucial elements for our example. Based on the above upper-level structure now we can try to model the same concepts from Example 1, but referring to a new structure (see Figure 19). This description does not contradict UDDI semantics, because *keyedRefernce's tModelKey* referring to a property assumes that all *keyValues* of *keyedRefernces* with the same *tModelKey* belong to the set of objects referred by this property.

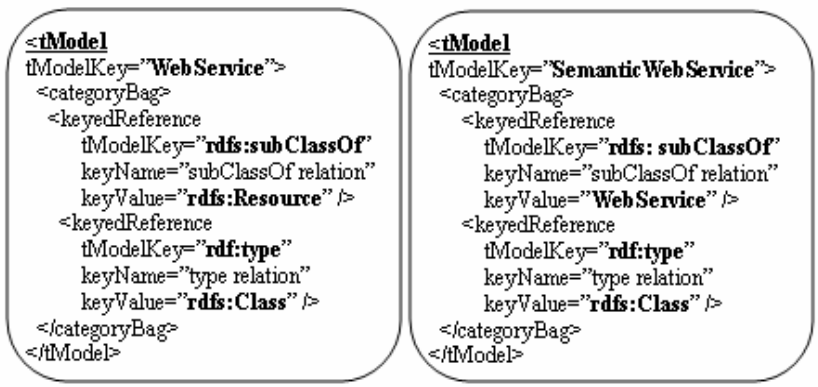

Figure 19. Definition of classes

7. CONCLUSIONS

The main goal of this paper was the evaluation of UDDI capabilities to store semantic descriptions of entities to enable semantic discovery and their integration with Semantic Registries in future based on mature and widely adopted UDDI specifications. Our conclusion is that UDDI as such provides enough support for registration of semantically annotated resources, but additional efforts are needed to elaborate API to support a semantic discovery of registered resources.

We have presented an approach of mapping RDFS upper concepts to a UDDI data model using a tModel structure.

While other publications in the area of enhancement of UDDI with semantics consider mainly semantic discovery and relevant enabling architectures, we have an opinion that the challenge of publishing semantic annotations of resources in UDDI has to be met without changing the UDDI architecture and APIs to enable semantic queries.

We consider two on-going projects (ASG and SmartResource) as the use cases and we have shown that publishing an ASG service and a domain

ontology into UDDI can be performed based on mapping WSMO to a UDDI information model. The SmartResource project could use UDDI to implement Notice Boards for registering semantically annotated resources in a P2P environment. In parallel to a practical utilization of UDDI in these two research projects, further research is needed to elaborate semantic discovery algorithms and APIs of UDDI based on the proposed way of storing semantics in UDDI.

8. ACKNOWLEDGEMENT

This research has been supported partly by the "Proactive Self-Maintained Resources in Semantic Web" (SmartResource) project funded by TEKES and the industrial consortium of Metso Automation, TeliaSonera, TietoEnator and Science Park and partly by the "Adaptive Services Grid" (ASG) 6th Framework Integrated Project (EU-IST-004617).

9. REFERENCES

[ASG] "Adaptive Services Grid", Integrated project supported by European Commission, http://asg-platform.org/

[DAML] "DAML Language", http://www.daml.org/language/

[DAML-S] "DAML Services", http://www.daml.org/services/

[FIPAAA] "FIPA Abstract Architecture Specification", http://fipa.org/specs/fipa00001/

[FIPAAM] "FIPA Agent Management Specification", http://fipa.org/specs/fipa00023/

[Kaikova2004] Kaikova H., Khriyenko O., Kononenko O., Terziyan V., Zharko A., Proactive Self-Maintained Resources in Semantic Web, Eastern-European Journal of Enterprise Technologies, Vol. 2, No. 1, 2004, pp. 4-16.

[Kaykova2005] Kaykova O., Khriyenko O., Kovtun D., Naumenko A., Terziyan V., Zharko A., General Adaptation Framework: Enabling Interoperability for Industrial Web Resources, In: International Journal on Semantic Web and Information Systems, Vol. 1, No 3, 2005, Idea Group, pp. 30-62.

[Kawamura2004] T. Kawamura, J. D. Blasio, T. Hasegawa, M. Paolucci, K. Sycara, "Public Deployment of Semantic Service Matchmaker with UDDI Business Registry", Proceedings of 3rd International Semantic Web Conference (ISWC 2004), LNCS 3298, pp. 752-766, 2004.

[Kawamura2003] T. Kawamura, J. D. Blasio, T. Hasegawa, M. Paolucci, K. Sycara, "Preliminary Report of Public Experiment of Semantic Service Matchmaker with UDDI Business Registry", Proceedings of First

International Conference on Service Oriented Computing (ICSOC 2003), LNCS No. 2910, pp. 208-224, 2003.

[Moreau2005] L. Moreau, S. Miles, J. Papay, K. Decker, T. Payne, "Publishing Semantic Descriptions of Services", Semantic Grid Workshop at GGF9, 2005.

[OWL] "Web Ontology Language", http://www.w3.org/2004/OWL/.

[OWLS] Ontology Web Language for Web Services (OWL-S) 1.0, http://www.daml.org/services/owl-s/1.0/.

[Paolucci1] M. Paolucci, T. Kawamura, T.R. Payne, K. Sycara, "Semantic Matching of Web Services capabilities", Proceedings of First International Semantic Web Conference (ISWC 2002), IEEE, pp. 333-347, 2002

[Paolucci2] M. Paolucci, T. Kawamura, T.R. Payne, K. Sycara, "Importing the Semantic Web in UDDI", Proceedings of E-Services Semantic Web Workshop (ESSW 2002), 2002.

[Pokraev2003] S. Pokraev, J. Koolwaaij, M. Wibbels, "Extending UDDI with Context-Aware Features Based on Semantic Service Descriptions", Proceedings of the International Conference on Web Services, ICWS '03, June 23 - 26, 2003, Las Vegas, Nevada, USA. CSREA Press 2003, ISBN 1-892512-49-1, pp. 184-190.

[Powers2003] Shelley Powers, Practical RDF, O'Reilly, July 2003, ISBN: 0-596-00263-7, 350 pages.

[RDF] "Resource Description Framework", http://www.w3.org/RDF/.

[RDFS] "RDF Vocabulary Description Language 1.0: RDF Schema", http://www.w3.org/TR/rdf-schema/.

[RuleML] "The Rule Markup Initiative", http://www.dfki.uni-kl.de/ruleml/.

[SemanticWeb] "Semantic Web", http://www.w3.org/2001/sw/.

[SmartResource] "SmartResource: Proactive Self-Maintained Resources in Semantic Web" project, http://www.cs.jyu.fi/ai/OntoGroup/SmartResource_details.htm

[Srinivasan2004] N. Srinivasan, M. Paolucci, K. Sycara, "An Efficient Algorithm for OWL-S Based Semantic Search in UDDI" Semantic Web Services and Web Process Composition, First International Workshop, SWSWPC 2004: 96-110.

[UDDIspec] UDDI Specifications, http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm.

[UDDI] "Universal Description, Discovery and Integration", http://www.uddi.org/.

[WSMO] "Web Services Modeling Ontology", http://www.wsmo.org.

[WSMOReg] "WSMO Registry", Working Draft, http://www.wsmo.org/2004/d10/v0.1/.

[WSDL] "Web Service Definition Language", http://www.w3.org/TR/wsdl.

[XML] Extensible Markup Language specification site, http://www.w3c.org/XML/.