International Journal of Spatial Data Infrastructures Research, 2008, Vol. 3, 168-191

An extensible semantic catalogue for geospatial web services*

Patrick Maué
Institute for Geoinformatics, University of Münster
pajoma@uni-muenster.de

Abstract

Searching for web services is a challenging task due to the diversity of existing discovery tools. Varying standards for service descriptions require different tools to query their content. These standards exist for good reasons. Web services are used in different application environments and special solutions are required for their integration. We suggest an architecture for web service catalogues, which takes this diversity of standards into account. Its extensibility allows for a complete separation between service discovery and service description. Using ontologies to model specific knowledge within the service descriptions enables the support of different description standards without loosing their specific advantages. The discussion and implementation focuses on geospatial web services. Due to the diversity of representations for spatially referenced data, different specifications exist for them, making the problem of different standards for service discovery and service description most evident here.

Keywords: Geospatial Information Retrieval, Semantic Web Services, Web Service Catalogues

1. INTRODUCTION

Geospatial information is a complex product. Due to the dynamic character of the phenomena represented, regular updates are required to keep it consistent (Longley, Goodchild et al. 2005). Different representation forms exist to accommodate the diversity of the geographic objects' spatial characteristics, captured either in raster or vector based data. The INSPIRE Directive (CEC 2007) for a European Spatial Data Infrastructure (SDI) specifies the usage of network services for the sharing of geospatial data. Service implementations have to comply with agreed specifications to ensure syntactical interoperability between the different participating members (Paragraph (17) in CEC 2007). Web

DOI: 10.2902/1725-0463.2008.03.art10

services (Erl 2005) are one option to achieve the desired separation between the data management and the tools which, for example, use the data to display it on a map. They are based on well-defined standards, which make them the technology of choice for complex and heterogeneous networks like SDIs. Other benefits, for example the possibility to limit the covered area of the often expensive geospatial data, contribute to the growing acceptance of this technology for Geographic Information Retrieval (GIR). The integration of web services at the client‘s side, to add for example their results as a new layer to a data set within a Geographic Information System (GIS), requires knowledge about the service location. Catalogues, acting as brokers between data provider and data requester (Erl 2005), provide a convenient way for the discovery of such location. Clients search for web services which match specific search criteria or browse registered web services stored in the catalogue’s repository.

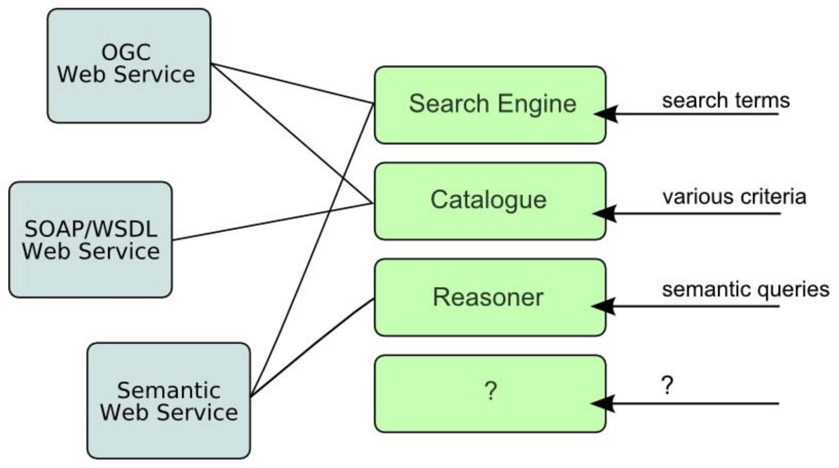
Discovery relies on service descriptions. Such metadata about web services contains information about the functionality and additional properties, e.g. keywords for categorisation of contact information of the service provider. How easy it is to find a web service (the discoverability) depends on the amount of information included in the associated service description (Erl 2005). The more fully characteristics of web services are described, the more elaborated search queries become possible with subsequent higher precision of the results. Due to different purposes of web services, multiple standards for service descriptions with varying levels of information exist. Metadata about geospatial web services, for example, include information about the spatial qualities of the data provided. A catalogue supporting such web services is able to process requests containing spatial search criteria like the restriction that resulting web services should only serve data covering a specific region. The Open Geospatial Consortium[1] (OGC) proposed several specifications for geospatial web services to standardise the discovery and execution. More generic standards developed by the World Wide Web Consortium[2] (W3C) are used for other (non-spatial) web services. Other proposals, for example for Semantic Web Services, are emerging (Staab, van der Aalst et al. 2003) or are expected to evolve in the near future. Web service discovery has to take this into account as well.

Figure 1 illustrates some of the discovery tools available and the resulting heterogeneity problems. Web services compliant to OGC or W3C specifications are usually registered to catalogues. Catalogue specifications with different discovery capabilities exist: the Universal Description, Discovery and Integration (UDDI) protocol for WSDL-based web services (Curbera, Duftler et al. 2002; OASIS 2004) and the OpenGIS Catalogue Service (OGC 2005) for OGC Web Services are commonly used. Keyword-based search engines additionally support searching for indexed service description documents with the help of

[1] About the OGC: http://www.opengeospatial.org/ogc
[2] About the W3C: http://www.w3.org/Consortium/about-w3c.html

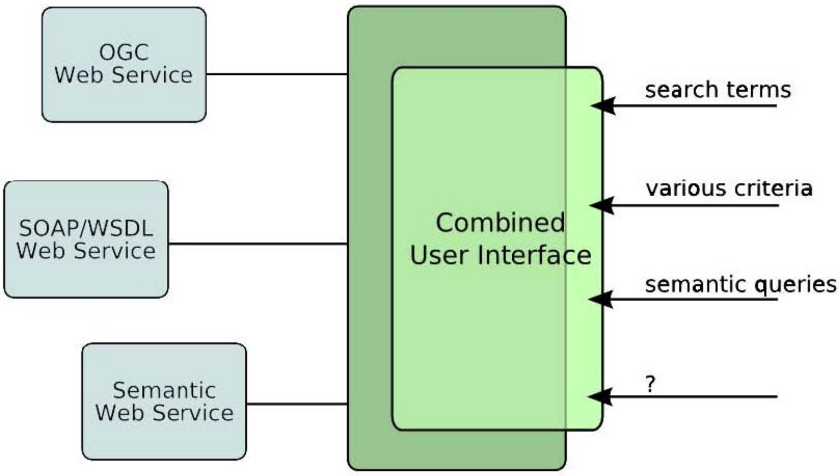
search terms. Semantic Web Services make a far more sophisticated discovery available by using logical reasoning to infer the similarity between semantic queries and ontology-based service descriptions (Sivashanmugam 2003). Due to new technology developments, additional solutions for web service discovery are expected to emerge in the future (Staab, van der Aalst et al. 2003).

Figure 1: Multiple options for web service discovery



The existence of different standards for service descriptions and discovery tools renders integration of web services difficult. The wide range of possibilities to search a web service makes the discovery ambiguous. A single discovery tool can not cover all existing web services. Search engines index only parts of the Internet, catalogues contain only web services which have been registered by the service provider. Thus, searching a web service becomes tedious by requiring the usage of multiple discovery tools to assure a sufficient number of potentially relevant web services. Each discovery tool is constrained to specific queries. This dependency renders service discovery using only one of the tools incomplete. Searching geospatial web services with a search engine only addresses the textual properties of the service description, not the spatial qualities. Catalogues allow for more sophisticated queries, but depend on the associated service description standard. Service descriptions accessible through different discovery tools can cause inconsistencies due to the redundant storage of the service description. Changes of the original web service might not be forwarded to all the discovery tools where the outdated service description has been registered.

Figure 2: Semantic catalogue as proposed solution



We propose an extensible architecture for a web service catalogue which supports multiple service description standards and discovery tools. It provides a solution for the problems identified without loosing the unique advantages of standards which define the extent of needed information in the service descriptions. A service-oriented architecture (SOA) based on modularised components makes the dynamic extension of its functionality possible. A logic-based formal language (see Section 4.2) has been selected for the persistent storage of registered web services as service ontologies.

Registration of a web service includes the translation of a service description into the corresponding service ontology and, optionally, the semantic annotation of the ontology with additional information about the web service. Figure 2 shows the benefits of the proposed catalogue. Service discovery is not ambiguous any more due to a single point of access. With the help of a reasoner it is able to interpret and compute different queries. The possibility to switch between sophisticated semantic queries, spatial queries, simple keyword-based queries, and more, enables the access to a greater variety of users. In the discussion about requirements for such a catalogue the achievement of completeness will be explained (see Section 3.4). Having only one repository avoids redundant storage of service descriptions and therefore the resulting inconsistent entries.

The paper continues with a review of the basic, common principles of service descriptions and the existing discovery tools. Section 3 discusses the requirements of the proposed architecture and is followed by a presentation of the implementation in Section 4. Section 5 takes the use case example introduced in the requirements section and shows a sample application of the implementation. Section 6 tries to put this work into context by comparing its results with similar approaches for web service catalogues. Before we come to a conclusion, future work is identified in Section 7.

2. PRINCIPLES OF SERVICE DISCOVERY

Web services within a SDI can be used for two reasons. They either act as remote data sources for local client applications. GIS software for example loads data from such web services to display current traffic conditions as layer on top of a map. Or their purpose is the processing of data, e.g. for the calculation of routes through a street network. Composition enables the interaction between such web services. A routing service might take the service providing the current traffic conditions as input to calculate alternative routes in the case of a traffic jam. The variety of functionality, the nature of the provided data, or even the targeted user community contributes to the reasons why different specifications for service descriptions and discovery tools exist. We have to distinguish between standards for metadata for web services in general and standards concerned about geographic information itself. The Dublin Core Standard[3] suggests a basic set of metadata entries to describe online resources. The ISO 19115 specification determines the extent and content of metadata for geographic information. Focused on geospatial web services some specific service descriptions standards are presented in the following section. Common properties of different service descriptions – which make an integrated discovery possible - are identified. At the end of this section, important discovery tool specifications are introduced.

The three steps required for integrating and composing web services are discovery, selection, and execution. Being discoverable depends on the availability of metadata about the web service‘s capabilities (Erl 2005). Queries contain search criteria which the web services requested have to satisfy. Discovery tools like catalogues process such queries, compare the criteria with the information available within the service descriptions of registered web services and return relevant records. The results include pointers to the original web service descriptions, which can be used by the client to execute the service. If the result comprises multiple web services, the most appropriate candidate has to be selected using evaluation methods. Several criteria support the user to infer the suitability of a service. Quality of Service (QoS) properties can, for example, be used to assess the reliability of a web service (Menasce 2002).

[3] About the Dublin Core Initiative: http://dublincore.org/about/

Executing a web service requires knowledge about its location, the operation syntax with its parameters, and the format of the expected results. Such properties are crucial for service execution. Missing or wrong values render the web service unusable. Every web service, no matter which specification has been used for the implementation, provides a service description including these functional properties. Service providers usually add supplementary metadata about the service context to improve its discoverability. A commonly used optional property is a list of keywords or a service title. A service requester makes use of this information for service discovery, for example to find all web services of the same service provider. Formalising more characteristics of a web service in the service description supports the assessment of its relevance during discover. Making it possible to extend web service metadata is therefore crucial to increase the efficiency of service discovery.

2.1 Schema-based service descriptions

Current service description standards like the Web Service Description Language WSDL (Christensen, Curbera et al. 2001) are based on the eXtensible Markup Language (XML). The available elements in such documents are predefined by the associated XML Schema, which determines the variety of information which can be stored in these documents. The WSDL format focuses on the primary functional aspects of service descriptions. Public operations and associated messages are described in an abstract manner and later linked to concrete network protocols. WSDL is widely used and forms, together with SOAP and the service registry UDDI (see below), the proposed standard of the W3C consortium. Choice of operation names or messages and their data types is not constrained. This makes WSDL-based service descriptions very flexible, but also renders the integration and composition more complex due to the implementation required on the client side. WSDL is a good choice to provide information about stateless web services used, for example, to process data. To reduce work for client implementation, other specifications standardise the access by predefining the names of the operations and the valid parameters. The *getCapabilities*-document used for OGC Web Services (OWS) is an example explained in the specification available at OGC (2005). It contains operations and message names similar to the WSDL document, but its focus lies on describing the data served.

2.2 Ontology-based service descriptions

XML-based service descriptions using fixed schemas contain sufficient information to execute a web service. Locating it on the other hand requires additional information about the service properties and capabilities. Switching from current schema-based service description standards, which are static concerning their number of allowed properties, to service descriptions based on formal languages makes service descriptions more flexible and more expressive.

Some of the problems tackled by Semantic Web Services are driven by different views on the data of the requester and data provider. Different background knowledge results in different terminology. A service provider from England might advertise a web service about quarries using the term “stone pit”, which is common in British English[4]. An American service requester would fail to locate it if the search works on keywords only. Synonyms and homonyms (the same word is used for different concepts) represent semantic heterogeneities which can cause the exclusion of otherwise web services during discovery (see Chapter 23 in Russell and Norvig 2002). In Probst and Lutz (2004), different types of such semantic heterogeneities are identified, and the usage of an ontology-based approach is proposed. An ontology is an “explicit specification of a conceptualisation” (Gruber 1995), which makes it possible to define concepts representing the features within a service description. The long-term goal of Semantic Web Services is an increasing automation of web service integration and composition so that, eventually, there is no need for any human interaction any more (Paolucci, Sycara et al. 2003). A human is able to detect ambiguous information and to decide from contextual information about the suitability of a web service. A software agent, on the other hand, depends on inference algorithms to derive new facts from existing concepts or to assess the similarity between different concepts.

Some of the languages used to specify ontologies are specialised for the use of modelling web service capabilities by providing a framework of predefined related concepts and relations. The W3C recommendation “Semantic Annotations for WSDL (SA-WSDL)” (Farrel 2007) is a light-weight approach which proposes semantic extensions for schema-based WSDL documents, making it possible to link output data types of service operations to concepts in an ontology. The OWL-S proposal (Martin, Burstein et al. 2004) suggests to avoid XML-based documents and rather use an ontology for web services based on the Web Ontology Language OWL (McGuinness and Harmelen 2004). For the implementation of the proposed semantic catalogue we use the Web Service Modelling Ontology WSMO (Feier, Dumitru et al. 2005) and the associated Web Service Modelling Language WSML. Reasons for this choice and its characteristics are further described in Section 4.

2.3 Using Catalogues for the Discovery

Searching for service descriptions in the context of a SDI relies on web service

[4] Searching WordNet (http://wordnet.princeton.edu) shows that "pit" and "quarry" are synonyms.

catalogues (Erl 2005). A catalogue manages a repository of service descriptions and provides access to it through interfaces. Abstract interfaces are used to enable interaction between other software applications and the catalogue. Graphical interfaces make it possible for a human user to browse through lists of registered web services or to search for specific web services. Searching requires the formulation of a query which contains a set of search criteria, e.g. the desired characteristics of the web service requested. The catalogue analyses the query, compares the given search criteria to the registered service descriptions and returns all matching records. The service requester is then able to evaluate these results by comparing and ranking the records returned and, eventually, to select the most appropriate entry.

The Universal Description, Discovery and Integration (UDDI) (OASIS 2004) registry is an example for a catalogue specialised for WSDL-based web services. The UDDI registry divides the available information into three different categories, each providing a separated search space. The first two categories include primary non-functional aspects, namely information about the service provider and the subjects covered. The third category enables the search for primary functional properties. UDDI provides simple facilities to append additional metadata, but the approach is inflexible and makes it tedious to add elaborated service descriptions. UDDI exclusively supports WSDL service descriptions (Curbera, Duftler et al. 2002) and the implementations are not able to manage web services compliant to different standards. It is possible to describe the capabilities of OGC Web Services with the help of WSDL (Sonnet 2005) and to register them in a UDDI registry afterwards. But geospatial web services require catalogues which support search queries with spatial features (Egenhofer 2002), for example to locate web services which provide data of a specific area. The OWS Cat specification (OGC 2005) is a guide for the implementation of a catalogue which is able to manage OGC Web Services and to compute queries with spatial search criteria. Similar to the UDDI, the OWS Cat works exclusively for OGC Web Services and is not intended to manage other web services. The use of discovery tools for ontology-based service descriptions is still an open issue scarcely discussed by the groups providing approaches like OWL-S or WSMO. In section 6 an extension of the UDDI is discussed which supports the discovery of extended WSDL documents using SA-WSDL.

We suggest an architecture for a web service catalogue which supports multiple service description solutions, schema-based as well as ontology-based. The implementation focuses on the discussed W3C and OGC compliant web services. Due to its extensibility, the architecture is able to provide different interfaces, depending on the context of the requester. A web interface includes facilities for querying and administrating the catalogue.

3. REQUIREMENTS FOR THE ARCHITECTURE

The previous section gave an overview of existing standards for service descriptions and the related discovery tools. The capabilities of web services are either described with the help of schema-based languages or extensible ontologies specified with the help of formal languages. Due to the support of spatial queries, catalogues are the preferred tool for the discovery of web services within a SDI. Different proposals, each adapted to its associated service description standard, exist for catalogues. The different standards require unique approaches to query their content. The varying options render service discovery and integration ambiguous and tedious (see Figure 1), but good reasons exist to use them. We suggest an architecture for a web service catalogue which supports this diversity of service description standards and discovery tools without loosing the specific advantages. The following section discusses the requirements for such an architecture.

3.1 From problems to requirements

We consider a simple use case for the identification of the requirements. Bob is working for a construction company and is responsible for providing sufficient supply of building material like gravel or sand for construction sites. Transportation of goods to the right place at the right time is a challenge. Storage of the required materials is usually not practicable. The building material, which comes from multiple quarries around the city, needs to be delivered at the time of consumption. The most suitable quarry, and accordingly the transportation routes for the trucks, varies from day to day due to changing prices. Web services provide the technology for communicating such time-critical information. Depending on the nature of the data required, means to search and access the necessary web service vary. Having multiple standards for service description and service discovery can lead to multiple problems. We identified three fundamental requirements which the catalogue architecture proposed needs to satisfy. It needs to be extensible to support different service descriptions types (section 3.2). Due to the demanded flexibility and the support of Semantic Web Services, the catalogue should be able to process ontologies (section 3.3). The third requirement is the completeness of discovery. Queries asking for primary aspects should cover all registered web services, independent from the original standard (section 3.4).

3.2 Extensibility

To realise the requirement of supporting different service types, the functionality of the catalogue needs to be extensible. This is achieved by utilizing a modularized infrastructure based on a service-oriented architecture. The different modules are self-contained software components which can be deployed into the semantic catalogue at runtime. The functionality of such components is only accessible through services, which are defined on interface level as sets of operations. Other components can bind such services by locating instances of the interface requested in the application environment and calling the service operations at runtime. This loosely-coupled discovery and execution allows for removing or changing modules without breaking dependencies or affecting the functionality.

3.3 Semantic enablement and persistent storage

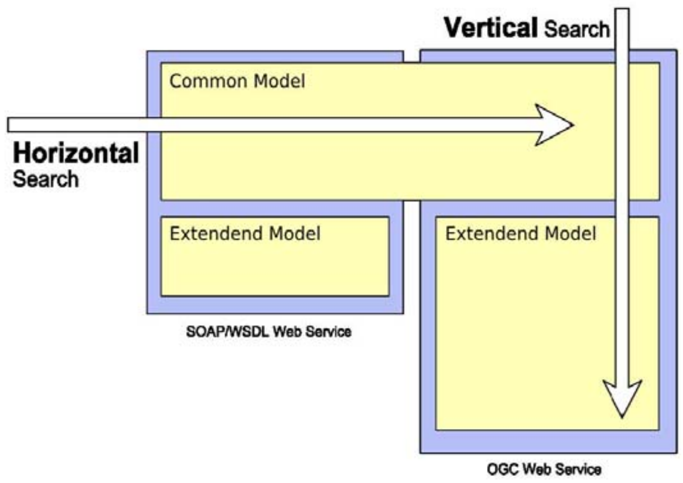
The service descriptions within the catalogue require persistence, preferably by utilising a database management system. Such systems provide a scalable way for consistent storage, especially in a multi-user environment. This implies the serialisation of the service descriptions into a common storage format. The format has to be expressive enough to support different service description languages. To support Semantic Web Services, service ontologies encoded in WSML are used for persistent storage. Formal languages provide the required expressiveness to cover the information available in schema-based service descriptions as well as the more sophisticated information within ontology-based service descriptions. The service descriptions are transformed into the corresponding WSML representation, the service ontology, and stored in its serialised form. While this is feasible for syntactic service descriptions, it is a complex task for semantic service descriptions like OWL-S. If the formal language used for the imported semantic service description is more expressive than WSML (in particular its WSML flight variant which corresponds to First Order Logic), a loss of information can not be avoided during the transformation into its corresponding service ontology.

Service descriptions encoded in formal languages can be interpreted and processed by generic reasoning engines. Linking the local concepts in these service descriptions, or service ontologies, to more generic domain knowledge formalized in domain ontologies opens up new possibilities to improve the discovery experience. The linking is also called semantic annotation, and is further described In Klien, Fitzner, et al. (2007). Allowing for more precise search queries expressed in the same formal languages makes semantic query processing possible. A reasoning algorithm can then match query and registered service descriptions on the semantic level and further filter out irrelevant records.

3.4 Completeness of discovery

To achieve the requested completeness of discovery, a mapping between service descriptions of different service types needs to be accomplished. Service discovery requesting specific properties common for all registered web services has to be supported as well as service type specific inquiries which affect only a subset of the records. The primary aspects of a web service comprise the functional properties and a fixed set of metadata attributes, which is a subset of the Dublin Core metadata standard for online resources. The secondary aspects include additional metadata as well as secondary functional properties. The import process transforms all service descriptions into a service model, which comprises the common model capturing the primary aspects and the extended model including all additional information. Completeness is achieved by distinguishing between horizontal and vertical search modes (see Figure 3). Horizontal search works solely on the common model, which is mandatory for each service. A query containing search terms for the name of the service provider would be an example. Service type specific queries, including, for example, the constraint that resulting web services need to provide data with a specific spatial feature, affects only web services whose service descriptions contain such information. To make use of such specific capabilities, a vertical, service type specific search mode needs to be supported.

Figure 3: Different search modes make completeness of discovery possible



Different search modes require different user interfaces, which support the service requester to formulate search queries. Providing input forms for search terms like keywords seems to be simple, but including spatial features renders the design more challenging. Maps can be utilized to enable the selection of a coordinate or to draw rectangles which represent the requested areas. Writing semantic queries, for example in WSML, requires sophisticated knowledge about formal languages and is usually too challenging for the service requester. Wizards can be used to hide the complexity and automatically construct semantic queries from the user input. The proposed catalogue architecture includes a component with a web interface (see Section 5) which helps to formulate simple queries and the possibility to attach additional interfaces. A component, which supports additional service description standards, has to provide the corresponding service type specific query and import interfaces.

4. IMPLEMENTATION

The requirements identified in the previous section address the problems coming with web service discovery. This section presents an implementation of the architecture proposed. It does not include all aspects discussed as requirements, but is focussed on the specific needs identified for the use case. The catalogue implemented supports the import of *getCapabilities*-documents of OGC Web Feature and Web Mapping Services. A web interface provides the necessary query and administration tools. The query interface enables users to formulate simple queries with keywords, spatial queries, and more sophisticated semantic queries. The persistent storage is realized by transforming the service model into service ontologies and saving their serialized representations into a database.

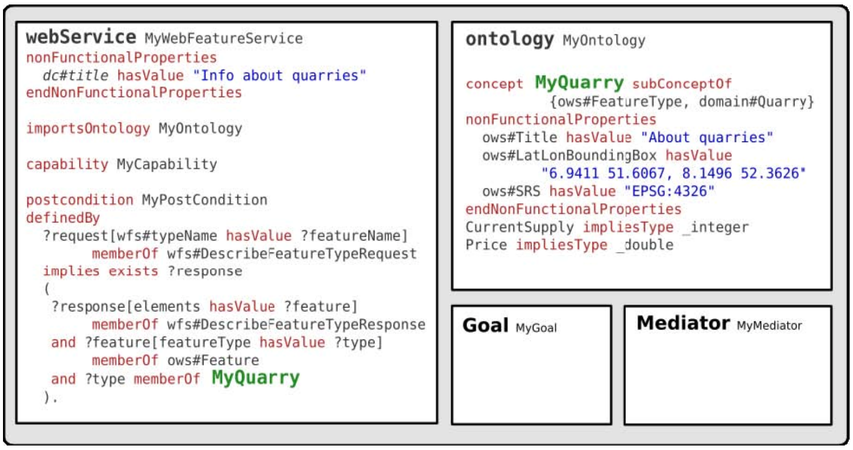
4.1 Implemented components

A modularized, service-oriented architecture helps to accomplish the required extensibility of the catalogue. The Service Platform proposed by the Open Service Gateway Initiative (OSGI) has been selected for this task. OSGI applications are made of collections of components, namely the platform runtime and multiple bundles with the required functionality. The runtime is responsible for the bundle management and the allocation of the bundle context. It invokes other bundles by calling the *start()*-method in the included bundle activator. The location of this activator is defined in the bundle descriptor file, which is, together with optional third party libraries and the actual logic, the content of a bundle. The descriptor file can include metadata about the bundle, e.g. the name of its provider. Services are used for the communication between bundles and are defined as interfaces which have to be registered to the bundle context. This context is managed by the runtime and is also used for the discovery of services by other bundles. For example, one bundle used in the catalogue provides logging functionality. On start-up the activator registers an implementation of the *LogService* interface to the bundle context. Other bundles which require event logging can then use this implementation by searching for services within the bundle context implementing this interface. Accessing the bundle functionality is only possible through the use of the operations defined in the associated service interfaces. Making the catalogue extensible by allowing different bundles to provide the same functionality, e.g. importing a service description, requires a set of predefined service interfaces on which these bundles have to agree.

4.2 Introduction to WSMO

The support of different service description standards requires the transformation of the parsed original schema-based service descriptions like the *getCapabilities*-document of OGC Web Services into service ontologies like the WSMO. Note that the presented approach is not limited to the WSMO framework. For this implementation WSMO/WSML has been chosen due to its ease of use, availability of tools (though more and better tools are available for more common languages like OWL), and high expressiveness.

Figure 4: Example of a service ontology in WSML



As shown in Figure 4, WSMO comprises four sections: *WebService*, *Ontology*, *Mediators* and *Goals*. The primary aspects of web services are part of the *WebService*. The *capability* of a web service is defined in terms of *preconditions* and *assumptions* which have to be valid before invoking the service and *postconditions* and *effects* afterwards. The *interface* section contains the *orchestration* and the *choreography*, which is used to relate operations defined in the capability section to actual service endpoints. If the execution of a web service relies on a composition of additional web services, its configuration is specified in the orchestration. Concepts within the capabilities section, like the concept "MyQuarry" in Figure 4, are defined in the associated ontology. Mediators provide a scalable way to link heterogeneous resources, for example for the integration of external ontologies. Goals represent discovery requests and are matched to service descriptions by a reasoner to assess similarity.

Having a catalogue with enhanced reasoning capabilities allows for semantic query processing. It matches the queries (the WSMO goals) to the registered service descriptions (WSMO WebServices). The reasoning engine interprets the ontology's assertions and is able to infer if, for example, a query concept matches a concept from the service ontology. A more thorough introduction into logical inference is given in Russel and Norvig (2002) The implementation presented is not making use of a reasoning engine; at the time of writing a stable release has not been available yet. The presented approach was further developed in the SWING project[5], here the reasoning capabilities of the WSMX platform were used.

5. USE CASE WALKTHROUGH

After having introduced the use case, the inferred requirements and the resulting implementation, this section answers the question of how Bob is able to use the semantic catalogue to find the information sources he needs for the delivery task. Although semantic query processing was not implemented, we discuss the potential benefits by showing an example of a semantic conflict between a query and a web service. The screenshots in this section can only show a small part of the user interface implemented. The accompanying video[6] provides a more detailed discussion of the possible query capabilities of the implementation.

Locating suitable quarries in a reasonable distance to the construction sites assumes a query which includes the following two search criteria: (1) the search term "quarry", because the returned web services should include information about quarries and (2) the spatial constraint that the quarries have to be within a certain distance to a given location. The implemented catalogue supports such thematic and spatial aspects by combining them to a semantic query in form of a WSMO goal. The query wizard is based on predefined goals and dynamically incorporates already entered thematic or spatial search criteria. A keyword, for example, appears in the corresponding keyword field within the non-functional properties. The user is able to modify the predefined goal to add additional criteria, which the requested web service has to satisfy.

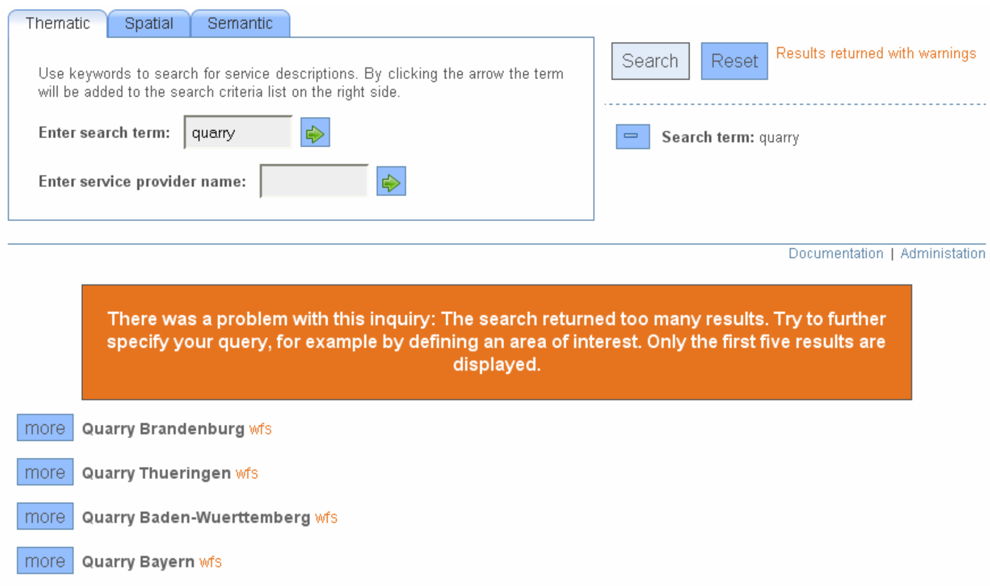
Bob does not have to enter all criteria at once. The user interface allows for adding and removing criteria, which makes it possible to approach the optimal

[5] More information at: http://www.swing-project.org
[6] Additional material can be found at: http://ifgi.uni-muenster.de/~pajoma/skat

query in multiple steps. Taking the required information of the locations of quarries as example, Bob first enters the term "quarry" as keyword into the corresponding input field at the thematic tab.

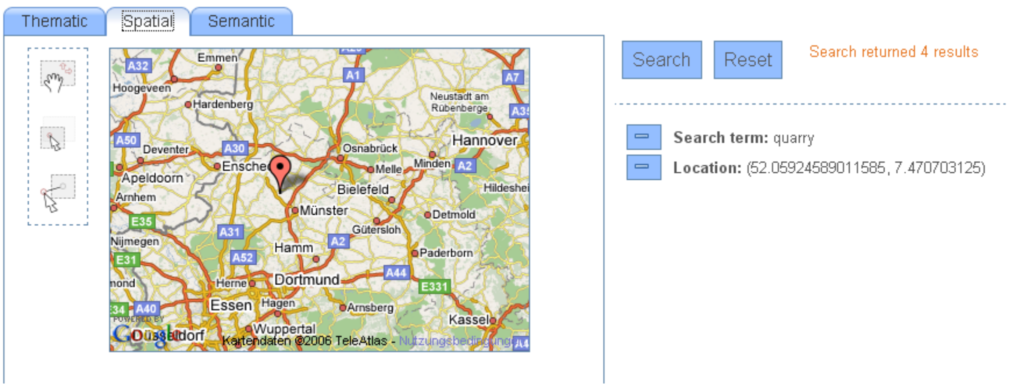
As shown in Figure 5 the catalogue returns all registered web services which have a matching term either in any title, description or keywords section. Using at first only keywords gives Bob a first impression of the possible results. In this case, the number of matching web services exceeds a certain threshold. This produces a warning which asks for reformulating the query by specifying additional search criteria.

Figure 5: Screenshot - First attempt by using only search terms



Bob is only looking for web services which provide the required information for a specific region. All web services which provide the required information but cover different areas are therefore not of interest. Contrary to thematic queries, spatial queries do not wrongly filter out relevant web services due to semantic heterogeneities. An inquiry could therefore just as well start with the definition of the spatial query. After switching to the spatial tab (see Figure 6) Bob is able to specify the location of the construction site by pointing on a position on the map. The resulting query combines the identified thematic and spatial features by integrating the criteria into a predefined WSMO goal.

Figure 6: Screenshot - Adding a spatial feature to filter out irrelevant web services.

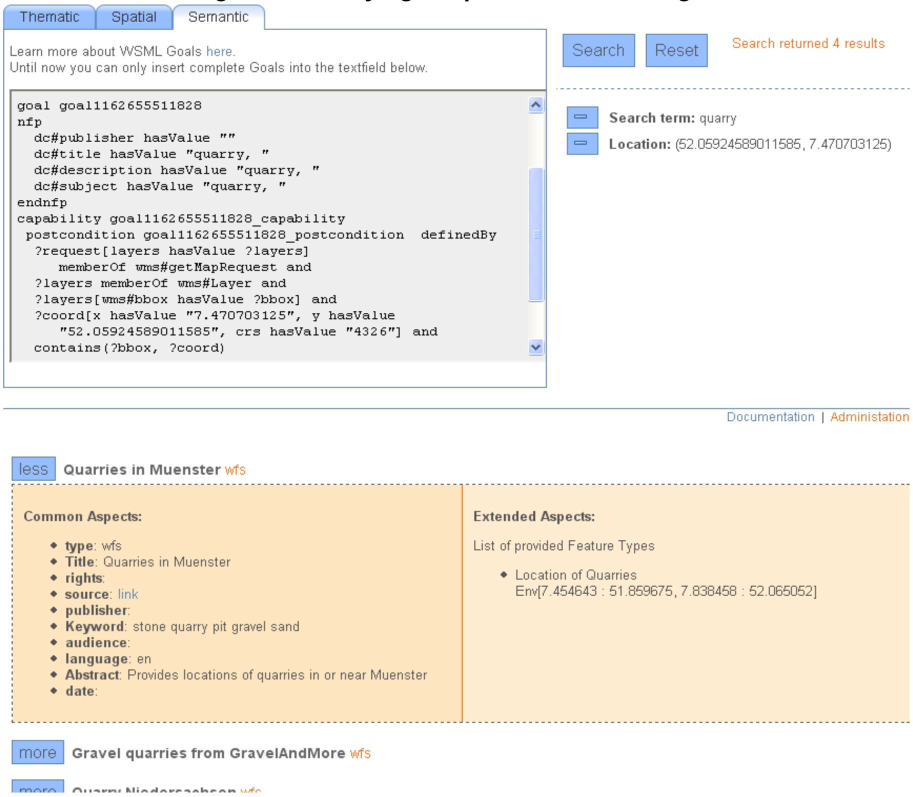


Adding search criteria to the query helps to raise the precision of the search results, because irrelevant web services get filtered out. However, relying on search terms only is problematic. Due to semantic heterogeneities or the lack of elaborated service descriptions, formerly relevant web services might get excluded. Not relying on keywords, but using a semantically-extended query can help here. Switching to the semantic tab (see Figure 7), Bob has the possibility to modify the WSMO goal directly. Of course, direct modification of the goal requires expert knowledge about the formalisms used in WSML. Better visualisation techniques for ontologies, and better user support to construct semantic queries using these ontologies, is subject to further research.

Bob could state that he is looking for features which represent Quarries with the following (simplified) expression: *?quarry memberOf domain#Quarry*. In the domain ontology, the domain experts modelled different concepts as similar to a quarry, namely excavation sites, exploration sites, pits, and more. The reasoning engine would infer that Bob is also looking for web services linked to, for example, the concept of an exploration site. The result returns more relevant web services. And on the other hand Bob didn't have to use more keywords, which reduces the risk to have either too imprecise queries, or too many web services filtered out in the search.

In comparison to more traditional discovery tools like search engines or service type specific catalogues, Bob's situation has been improved considerably with the use of the semantic catalogue proposed. The number of potentially registered web services has been increased significantly due to the support for multiple service types. The implementation supports Web Map Services, Web Feature Services, and Semantic Web Services with service descriptions written in WSML. With the extensible query interface proposed, Bob has been able to formulate a semantic query which included thematic as well as spatial search criteria. Queries allow for a more precise matching between the user queries and registered web services.

Figure 7: Modifying the predefined WSMO goal.



A higher number of registered web service leads to potentially more relevant web services, and more sophisticated search queries can result in more precise search results. The planned reasoning capabilities of the catalogue will help to increase the number of potentially relevant web services even more, making a semantic catalogue of the kind presented here the discovery tool of choice for the discovery of heterogeneous information sources.

6. RELATED WORK

Separating web service discovery from the standards used for service descriptions is not a novel approach. This section shows other proposals which try to eliminate borders between multiple standards for web service messaging, description and discovery. Semantic Web Services are a relatively new technology and solutions which integrate the established schema-based with the emerging ontology-based service descriptions are therefore rare. Another distinction can be made regarding the approach. Some proposals, including the semantic catalogue discussed here, require a transformation of the source description documents into an internal model. Restricting the functionality to act solely as an adapter between different discovery mechanisms could keep the source descriptions untouched.

ebXML, the *Electronic Business using eXtensible Markup Language*[7] , is a family of XML-based standards which helps to improve the communication in electronic business. As UDDI, ebXML is used to register and advertise web services. But ebXML includes increased flexibility, which simplifies the annotation of existing documents. Building a catalogue for web services which transforms and stores the service description documents in the ebXML format could provide an approach to support multiple service description languages. The *Red Spider Catalog2* (Erdas 2004) utilizes ebXML to handle the extensive information within the *getCapability*-document as well as additional metadata documents provided by OGC Web Services. It is compliant to the OGC Catalogue Services Specification (OGC 2005), but does not include other means to query the web services registered, for example with the help of semantically enriched queries.

The WizNet (Dustdar and Treiber 2004) architecture tries to integrate ebXML as well as UDDI and is therefore an example of an approach which takes multiple discovery mechanisms into account. WizNet links the formerly incompatible web service registries by acting as adapter between the client and the registries. It relies on web service profiles, the so called Views, which provide an abstracted view of service descriptions independent from the underlying registry. But the service descriptions used in ebXML or UDDI registries are still written in XML-based languages, thus there is no support for Semantic Web Services and reasoning.

A proposal which integrates Semantic Web Services is the METEOR-S Web Service Discovery Infrastructure (MWSDI, see Verma, Sivashanmugam et al. 2005). Its primary goal is scalability and most of the work has been done in the conceptualisation and implementation of a peer-to-peer (P2P) based prototype. Improving scalability through P2P techniques is an important issue which is also discussed in the following section about future work. MWSDI supports web services described and annotated in WSDL-S (Akkiraju, Farell et al. 2005), a proposal to semantically enrich WSDL documents. WSDL-S has been, by now, succeeded by the already introduced SA-WSDL recommendation. As for WSDL, UDDI is used for the publication and discovery of the WSDL-S documents. By relying on established standards like WSDL and UDDI and not inventing new

[7] About ebXML: http://www.ebxml.org/geninfo.htm

solutions like OWL-S or WSMO, the people behind METEOR-S expect a higher acceptance and a smooth migration to Semantic Web Services. Several points of interoperability have been identified between the METEOR-S project and WSMO (Verma, Mocan et al. 2005). Due to its extensibility, a semantic catalogue built using the proposed architecture could be able to import WSDL-S enriched service descriptions and provide the required discovery mechanisms.

The main focus of the architecture proposed is to provide means to publish and discover web services independent from the chosen service description language and discovery mechanism. Related work exist which tries to bridge the gap between specific standards, but no project has been identified which is able to support any standard.

7. POSSIBLE EXTENSIONS

The implementation discussed in Section 4 is focussed on components which support the use case. The semantic catalogue is based on an extensible infrastructure, thus its functionality can be simply enhanced by plugging in new components. Compatibility to the catalogue is realised by implementing the service interfaces introduced in Section 4.1. This Sectionpart discusses possible extensions of the already existing implementation, for example to increase the precision of search results.

Relying on a single instance of the catalogue can be the cause of multiple problems. The algorithms used to compute requests, transformations, or the reasoning are complex and have a negative impact on the catalogue’s performance. Connecting multiple instances of the catalogue through a peer-to-peer (P2P) network helps to distribute the load and to avoid this bottleneck problem. Each peer is an instance of the catalogue with a repository of registered service descriptions. Replication of the repository content to other peers enables the network to cope with failure of single nodes without loosing any information. Discovery within a P2P network spans over all catalogues available in the network, which would satisfy the demanded completeness of discovery (see Section 3). Note that the approach presented is independent from the implementation, it can either be used in conjunction with distributed catalogues or for single instances.

Once a web service description is imported and semantically annotated, the service metadata within the catalogue repository has to be regularly compared to the original schema-based service descriptions. Web services tend to be unreliable, service interfaces might change due to new requirements. Changes render a catalogue record inconsistent, because it does not reflect the capabilities of the advertised web service any more. To avoid such problems, regular synchronisations are required to update the service descriptions. Such a mechanism could also allow for assessing the reliability and availability of the web service. This attribute is part of the Quality of Service (QoS) Parameter, which can be used for the evaluation of a web service. Additional QoS parameters include performance, security and other less often used service qualities. Extending the semantic catalogue by monitoring tools makes the calculation or inference of these parameters possible.

The question how to generate semantic annotations was briefly discussed in Section 3.3. After the import of a source service description like the *getCapability*-document into the semantic catalogue, the user is required to load the transformed service ontology into an ontology editor like WSMT to enable the mapping of application concepts to concepts at the domain level. Semi-automatic semantic annotation tools could simplify this task and might help to improve the average quality of the imported service descriptions. The algorithms included automatically identify the appropriate domain concepts and provide the user an interface to adjust the suggestions and improve the annotations generated. The nature of a suitable algorithm depends on the source material, in this case the service descriptions.

Search results usually include irrelevant entries. Reasons for this could be false or too general metadata, incorrect alignments during the semantic annotation, or too general search terms. To increase the rate of recall, the ratio between relevant results and relevant records in the database, several techniques have to be applied. Semantic heterogeneities can lead to exclusion of relevant web services. Properly annotated service ontologies in combination with sophisticated reasoning algorithms help to reduce these undesired exclusions. Furthermore, user feedback can help to identify relevant search results for specific queries. Learning algorithms (Russell and Norvig 2002) can utilize such feedback to assess the relevance of web services results. A combination of web services usually identified as relevant in former queries could have, for example, a higher probability to appear in future results as well. The precision of the search results, the number of relevant results compared to the overall result, depends partially on the accuracy of the query. The more search criteria the service requester specifies, the more non-relevant entries will get filtered out. But sophisticated search queries require elaborated metadata documents within the catalogue to avoid a decreasing rate of recall. Collaborative editing of metadata might help to improve metadata quality. Allowing trusted users to identify and correct false metadata entries could decrease the number of irrelevant entries in the search results.

8. CONCLUSION

The proposed architecture for an extensible semantic catalogue for web services satisfies the requirements discussed in Section 3. The functionality of the catalogue is dynamically extensible through the use of the OSGI service platform.

The transformation of service descriptions into their ontological representations makes reasoner-based discovery possible, which helps to avoid conflicts caused by semantic heterogeneities. Because simple queries affect only the common model, searching for example a provider name covers all registered web services. Complex queries with service type specific search criteria are supported as well, but are constrained to specific web services with the associated extended model. An implementation of the proposed architecture was introduced in Section 4. It makes the import and discovery of web services described either with WSDL or the OGC getCapabilities-document possible. WSMO has been used for the service ontology and the persistent storage.

Compared to a catalogue that is constrained to a specific schema-based standard for service descriptions, the architecture proposed has several advantages. The major benefit is the potential increase of relevant web services in the result set. This has been achieved by extensibility and semantic enablement. Extensibity makes the support of multiple service description standards possible. This can result in an increased number of registered and eventually relevant web services in the repository. With the transformation into service ontologies and the creation of semantic annotations, semantic heterogeneities can be reduced. Web services that were classified as irrelevant with classical search techniques might now be considered as a semantic match, raising the amount of resulting relevant web services as well.

A potential increase of the precision of the search results has also been achieved. Combination of multiple search criteria like keywords and spatial features makes more sophisticated matching between a query and the registered service descriptions possible. More elaborated search queries help to return web services which comply better with the assumptions of a service requester. Compared to current discovery tools, we posit that the proposed architecture for a catalogue makes a more efficient discovery and integration of web services available.

ACKNOWLEDGEMENTS

This research has been supported by the European Commission under the EU-IST Project No. FP6-26514 (SWING)

REFERENCES

Akkiraju, R., J. Farell, et al. (2005). Web Service Semantics - WSDL-S. *Proceedings of W3C Workshop on Frameworks for Semantics in Web Services*, Innsbruck, Austria.

Akkiraju, R., R. Goodwin, et al. (2003). A Method for Semantically Enhancing the

Service Discovery Capabilities of UDDI. Proceedings of IJCAI-03 Workshop on Information Integration on the Web (IIWeb-03). Acapulco, Mexico.

Christensen, E., F. Curbera, et al. (2001). Web Service Definition Language (WSDL). Available at http://www.w3.org/TR/wsdl (last access: June 2008).

Commission of the European Communities (2007). Directive 2007/2/EC of the European Parliament and of the Council of 14 March 2007 establishing an Infrastructure for Spatial Information in the European Community (INSPIRE).

Curbera, F., M. Duftler, et al. (2002). Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI". *IEEE Internet Computing* **6**(2): 86--93.

Dustdar, S. and M. Treiber (2004). WiZNet - Integration of different Web service Registries. Diploma Thesis.

Egenhofer, M. (2002). Toward the semantic geospatial web. GIS '02: *Proceedings of the 10th ACM international symposium on Advances in geographic information systems,* ACM.

Erdas. (2004). Product Sheet for Red Spider Catalog2. Available at http://www.ionicsoft.com/ (last access: June 2008).

Erl, T. (2005). *Service-Oriented Architecture (SOA): Concepts, Technology, and Design.* Prentice-Hall, Inc. Upper Saddle River, NJ, USA.

Farrel, J., H. Lausen (2007). W3C Recommendation: Semantic Annotations for WSDL and XML Schema. Available at http://www.w3.org/TR/sawsdl/ (last access: June 2008).

Feier, C., R. Dumitru, et al. (2005). Towards intelligent web services: The web service modeling ontology (WSMO). *Proceedings of International Conference on Intelligent Computing (ICIC'05)*. Hefei, China.

Freeman, E., E. Freeman, et al. (2004). *Head First Design Patterns.* O' Reilly & Associates, Inc.

Gruber, T. (1995). Toward principles for the design of ontologies used for knowledge sharing. *International Journal on Human-Computer Studies* 43(5-6): 907-928.

Klien, E., D. I. Fitzner and P. Maué, , (2007). Baseline for Registering and Annotating Geodata in a Semantic Web Service Framework. *Proceedings of the 10th Conference on Geographic Information Science (AGILE 2007)*, Aalborg, Denmark.

Longley, P., M. Goodchild, et al. (2005). *Geographic Information Systems and Science*. Wiley & Sons.

Martin, D., M. Burstein, et al. (2004). W3C Member Submission: OWL-S: Semantic Markup for Web Services. Available at http://www.w3.org/Submission/OWL-S/ (last access: June 2008).

McGuinness, D. L. and F. v. Harmelen (2004). W3C Recommendation: OWL Web Ontology Language Overview. Available at http://www.w3.org/TR/owl-features/ (last access: June 2008).

Menasce, D. A. (2002). QoS issues in Web services. *Internet Computing, IEEE* 6: 72--75.

OASIS (2004). UDDI Executive Overview: Enabling Service-Oriented Architecture. Available at http://uddi.org/pubs/uddi-exec-wp.pdf (last access: June 2008).

Open Geospatial Consortium Inc. (2005). OGC Catalogue Services Specification. Available at http://www.opengeospatial.org/standards/cat (last access: June 2008).

Open Geospatial Consortium Inc. (2005). OpenGIS Web Service Common Implementation Specification. Available at http://www.opengeospatial.org/standards/common (last access: June 2008).

OSGI Alliance. (2005). About the OSGi Service Platform - Technical Whitepaper Revision 4.0. Available at http://www.osgi.org/Links/Documents (last access: June 2008).

Paolucci, M., K. Sycara, et al. (2003). Delivering Semantic Web Services. International World Wide Web Conference (WWW2003), Alternate Paper Tracks.

Probst, F. and M. Lutz (2004). Giving Meaning to GI Web Service Descriptions. 2nd International Workshop on Web Services: Modeling, Architecture and Infrastructure, WSMAI 2004, Porto, Portugal.

Russell, S. and P. Norvig (2002). *Artificial Intelligence: A Modern Approach.* Prentice-Hall, Inc. Upper Saddle River, NJ, USA.

Sivashanmugam, K., K. Verma, et al. (2003).Adding Semantics to Web Services Standards. *Proceedings of International Conference on Web Services*. Las Vegas, Nevada, USA, CSREA Press.

Staab, S., W. van der Aalst, et al. (2003). Web services: been there, done that? *Intelligent Systems, IEEE* 18(1): 72-85.

Jérôme Sonnet (2005). OGC Discussion Paper: OWS 2 Common Architecture:

WSDL SOAP UDDI.

Verma, K., A. Mocan, et al. (2005). Linking Semantic Web Service Efforts - Integrating WSMX and METEOR-S. *Proceedings of the ICWS 2005 Second International Workshop on Semantic and Dynamic Web Processes (SDWP 2005*).

Verma, K., K. Sivashanmugam, et al. (2005). METEOR-S WSDI: A Scalable P2P Infrastructure of Registries for Semantic Publication and Discovery of Web Services. *Information Technology and Management* 6(1): 17-39.