A Mediation Framework for Web Services in a Peer-to-Peer Environment

Michael Mrissa
Université Bordeaux 1
Bordeaux, France
michaelmrissa@myway.com

Chirine Ghedira and Djamal Benslimane
Université Claude Bernard Lyon 1
Lyon, France
{djamal.benslimane,chirine.ghedira}@iuta.univ-lyon1.fr

Zakaria Maamar
Zayed University
Dubai, United Arab Emirates
zakaria.maamar@zu.ac.ae

Jacques Fayolle
ISTASE- Université Jean Monnet Saint Etienne
jacques.fayolle@univ-st-etienne.fr

Abstract

Centralized architectures based on the client server paradigm show their limits each time the Internet extends its capabilities. They present problems in terms of security, robustness, performance, high needs for administration. Short time solutions like load balancing and replication, or purchasing of more powerful hardware tend to cope with these limits for a while, but do not provide a definitive answer to these problems. Currently, different proprietary approaches hinder attempts to enabling communication between heterogeneous systems, while the Internet is developing toward a better interoperability with service-oriented architectures.

Recent open source and standardization efforts allow building a framework providing an original answer to interoperability and scalability over the Internet. The use of Web Services (WS) as a means of communication between information systems, combined with a Peer-to-Peer (P2P) architecture for a more reliable network resources repartition and decentralization, forms a stable base for a new kind of distributed architecture. Furthermore, the composition and aggregation of multiple WS allow providers to supply with large scale meta-services. However, the use of heterogeneous WS in a dynamic context like P2P still faces challenges concerning the matching of WS and their respective interfaces. This paper discusses the use of mediation as an alternative to deal with the constant evolution of Web Services availability, and proposes a framework for a WS-enabled P2P platform.

1 Introduction

Recent development of Web Services (WS) as a standard for interoperable WS-oriented platforms over the Internet is gaining momentum [2, 7, 9, 10, 14, 22]. The vision of WS as a software component allows to combine several WS, providing a global value-added WS, called composite WS. However, most existing works focusing on WS integration rely on user interaction for WS execution and composition, thus avoiding automation WS provisioning problems at these stages. In effect, essential knowledge enabling WS automation is either absent, like in architectures based on WSDL[1]/UDDI[2], or described in order to be interpreted by human beings [21]. Devices must be able to automate WS interoperation by interpreting their application domain and surrounding context. Consequently, it seems necessary to tend toward machine-understandable WS, using explicit WS semantics: this is the concept of semantic WS.

Ongoing research in the field of WS is moving toward automatic composition and aggregation of semantically described WS [1, 6, 11–13, 17, 18, 20, 21, 23–25]. WSDL is the standard specification describing WS interfaces. A semantic agreement represents a common starting point in order to interpret WSDL data in a particular domain or context. This agreement can be associated with an ontology[3], a vocabulary, or a set of rules in a standard representation. However, a WS composition based on a common agreement still requires mediation between the descriptions of composed WS interfaces. Indeed, data structures and contents received in a response to a message do not always match those required as an input for the next component WS. Output data from previous WS must be adapted in order to create one or many request messages with adequate parameters. However, in a dynamic environment like Peer-to-Peer (P2P), a static hard-coded mediation as when using languages like BPEL4WS[4]

[1] WSDL is a standard format describing WS endpoint, messages definition, and data bindings. http://www.w3.org/TR/wsdl

[2] UDDI provides WS lookup and discovery. http://www.uddi.org/

[3] An ontology defines the terms used to describe and represent an area of knowledge. http://www.w3.org/TR/webont-req

[4] BPEL4WS is a notation for specifying business process behavior

is not sufficient. When a component WS changes, there is a need for a semantic description of this WS interface in order to adapt new component WS of the composition.

At the same time, the dynamic autonomous nature of P2P architecture as a framework for building distributed systems is addressing the limits of client/server architectures. The P2P paradigm brings a relative independence regarding the architectural layer and the physical constraints of the underlying network [19]. This paradigm is based on a dynamic management of the network that allows continuous changes of network topology and resource availability. Moreover, considering each element of the system both as a client and a server allows solving problems like high load on a single server, replication and consistency between replicas, and lack of network resources in general. This decentralized approach naturally solves problems of resources centralization, like security sensible points and bottlenecks. Particularly, Verma et al. [25] present a P2P solution for a decentralized localization of UDDI registries. More and more information systems rely on a decentralized vision of the network, with P2P platforms like Java Jini (http://www.jini.org/), and JXTA (http://www.jxta.org/) open source projects, or proprietary approaches like Groove Networks (http://www.groove.net/). Following on previous work (Verma et al. [25], JXTA SOAP Project[5]), we consider that the distributed nature of P2P platforms allied with the standardized WS architecture, based on SOAP[6] and WSDL represent a powerful framework for building large scale interoperable and decentralized information systems.

Considering both the contexts of P2P and semantic mediation aforesaid, we focus in this paper on interface mediation problems related to composite WS in a dynamic environment. Our goal is to allow JXTA peers to compose standard external WS from the JXTA network, invoking each of the component WS in a generic way from its WSDL. At the same time, we aim at improving existing WS publication and discovery by using JXTA's architecture, basing our solution on previous research work [25]. The contribution of this paper is firstly a mediation component that uses semantic WS descriptions in order to automate WS composition, and a framework for WS integration into the JXTA model. The rest of the paper is organized as follows. Section 2 introduces the context of our work and reviews different approaches relative to a semantic enhancement to WS. Section 3 presents a running example and then proposes a classification of WS interface heterogeneities in a composition, before proposing a mediation solution for a dynamic composition of semantic WS. Then, section 4 outlines a framework for a semantic WS integration in a P2P environment. Finally, section 5 illustrates our framework, using the example introduced previously.

based on WS. http://www-106.ibm.com/developerworks/library/ws-bpel/

[5]http://soap.jxta.org/

[6]SOAP is the standard format of the transport layer for XML-encoded messages. http://www.w3.org/TR/soap/

2 Context and Related Work

2.1 Dynamic Context

In client server architectures, clients always rely on server resources, which are always granted available. At the contrary, in a P2P network topology, the availability of network resources like WS constantly change. This evolution leads to changes in the selection of component WS from an invocation to another. Furthermore, a WS may not be available at all at a particular invocation time. In this paper, we only consider the case when a composite service is available. Hence, the challenge is to find solutions that make an existing WS adaptable to new requirements of the composition. The use of mediation seems to be a reliable alternative to match WS [8]. In a static context, the semantics of the WS is known, therefore the mediation code remains static (for instance written in BPEL4WS). However, in a dynamic context, there is a need for an explicit semantic description of the WS, and this comes with the use of metadata[7].

There are four types of semantics for describing WS [25]: (i) functional semantics describe their action; (ii) QoS semantics give details about their characteristics like response time, cost, reliability and fidelity; (iii) data semantics describes their input and output interfaces; and (iv) execution semantics encompasses the idea of message sequence, conversation pattern, flow of actions, preconditions and effects of WS invocation. For the need of this paper, we only focus on data semantics. Hence, we study different ways to semantically describe input and output parameters between WS. However, it is important to stress that issues related to other types of semantics should be considered for real world applications.

2.2 Related Work

Semantic WS are at the convergence of two important fields of research which relate to technologies of the Internet: WS and the semantic Web. The semantic Web is interested in describing static information available on the Internet in a machine-understandable way (See Section 1). WS, as for them, are mainly concerned with inter-working between applications via the Web in order to make it dynamic.

The need for automating WS design and implementation processes joined the concerns at the origin of the semantic Web, namely how to describe knowledge formally so that

[7]Metadata is basically defined as "data about data", which is a set of assertions that describes all the features belonging to a particular domain.

it becomes machine-exploitable. Consequently, technologies and tools developed in the context of the semantic Web can enhance WS technology to bring relevant answers to automation problems. For example, the concept of ontology can play a dominating part in order to clarify WS semantics, and to facilitate human-to-machine and machine-to-machine communications.

2.2.1 OWL-S.

OWL-S is an ontology-based language used to semantically describe WS, focusing on their properties and capabilities [1]. The goal of OWL-S, built upon DAML-S[8], is to allow a better automation of WS-related tasks like discovery, invocation, composition and interoperation. Maamar et al. [16] summarizes the structure of OWL-S in a profile, providing appropriate information for WS discovery and matchmaking; a process model, representing the WS input and output parameters; and a Grounding section, specifying additional WS access information (like WSDL).

2.2.2 Annotating WSDL.

A way to semantically enhance WS description is to annotate WSDL files. Two main approaches are inline with this idea:

- With the help of the extensibility support of WSDL (V 1.2), files can be extended with semantic information [24].
- If using Java, the WS source code can be annotated, as usually WSDL files are issued from the WS code (Java2WSDL tool). Metadata descriptions for Java code components can be described in JSR 175[9]. In extension, JSR 181[10] provides WS specific metadata descriptions. The idea is to use those facilities to add semantic metadata in the WS Java source code.
- Other approaches map WSDL files into ontological constructs, using languages such as OWL-S [24].

2.2.3 Enhancing UDDI with Semantics.

The idea consists in registering semantic constructs in UDDI. UDDI data structure can be used to store semantic details of a service [24], by implementing tModels, which are metadata constructs that describe compliance with a specification, a concept or a shared understanding. The information contained in a WSDL file corresponds to the "grounding" section of a UDDI construct. It constitutes hence a building block of the description.

[8] DAML-S is a language defining a set of classes and properties specific to WS description. [1, 12, 13]

[9] [http://www.jcp.org/en/jsr/detail?id=175]

[10] [http://www.jcp.org/en/jsr/detail?id=181]

2.2.4 Other Approaches.

Two other approaches can be mentioned for semantically describing WS. A first one consists in using RDF in conjunction with WSDL, or some other standards, to describe a WS. This approach can be effective because it allows describing WSDL in RDF syntax, which makes it compatible with existing RDF-based systems [20]. A second approach aims at automatically generating metadata [11], using machine learning and clustering techniques in order to attach semantic metadata to WS (clustering techniques serving to categorize WS into the classifications included in WSDL). On another side, other groups [6, 14] are interested in defining a formal framework for allowing a better relation inclusion and understanding between the global properties of a composite service and local properties of its components. Their motivation is to develop techniques of composite WS properties checking and synthesis (construction), starting from the properties of its components.

3 A Semantic-Driven Approach for WS Mediation

3.1 Running Example

Throughout this paper, we use an example that is about constraints on resource availability. We consider that a user wants to find synonyms for French words, due to lack of French dictionary. In this case, three WS may be composed (Figure 1), proposing on the one hand a simple composition case, and on the other hand an example of answer aggregation. Indeed, two WS provide a list of synonyms for a given English word, and the third one is a translation service between French and English. Firstly the translation service

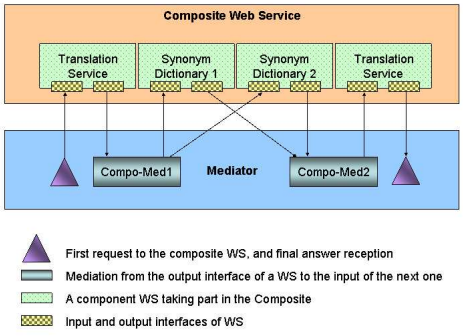


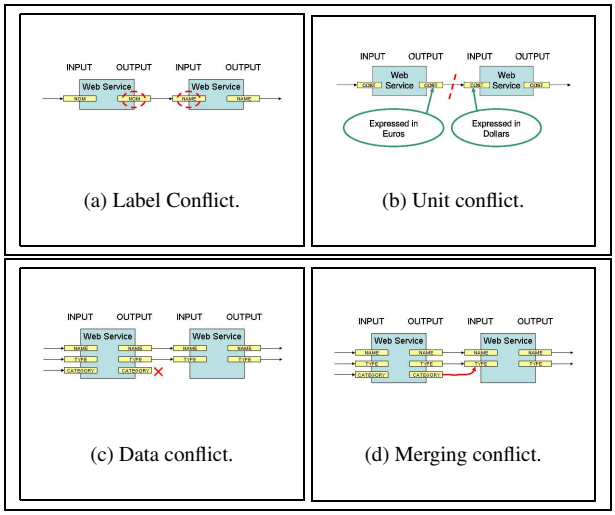
Figure 1. Composition of WS

converts the French word into English, then the two synonym WS are invoked with the English word and provide two lists of synonyms that are merged. Finally, each term

of this list is translated into French. The final answer is a list of French synonyms for the given French word.

3.2 Data Heterogeneity When Composing WS

This section classifies the heterogeneities that are raised when composing WS. However, this classification is not exhaustive, as it is mainly adapted from previous work [15]. In the context of WS, data conflicts can be described as follows:



(a) Label Conflict. (b) Unit conflict.

(c) Data conflict. (d) Merging conflict.

Figure 2. Conflicts classification

3.2.1 The semantic level.

- Label Conflicts.

Label conflicts (Figure 2(a)) reside in the vocabulary used for data description. For instance, a label conflict occurs when a French service uses the word ”NOM” in order to define a name whereas the English service uses the word ”NAME”. In this case, two different words have the same meaning. In another case, the same word may mean something different in another language.

- Unit and Value Conflicts.

Unit conflicts refer to different units used in a same description. For example a same “COST” field used as a service parameter can describe the cost of a translation, and be expressed in euros or dollars. This problem becomes more important with discrete data like temperature where units and scales can change (Celsius and Fahrenheit degrees for example). A value conflict occurs when different values have the same meaning. This is different from the “label” conflict described above because it concerns the content of a parameter and not its name. For instance the values used to address the type of a word can be “1”, “2” or “3” in a WS, and “m”, “f” and “n” in the second (Figure 2(b)).

3.2.2 The structural level.

Structural conflicts occur when parameters are structured in different ways while composing Web Services. In our example (Figure 2(c)), in the first service, a word is defined by a structure containing a “TYPE” field defining if the word is a noun, a verb or something else, a “CATEGORY” field telling if the word is male, female or neutral, and a “NAME” field containing the word itself. Thus, the service will have three input and output parameters. However, the next service only needs a “NAME” element containing the word and a “TYPE” field containing the type of the word (verb, noun, or else) as input parameters. We are so faced to a mismatch in the structure of the elements passed as parameters.

In figure 2(d) we present a particular case of structural conflict called a merging conflict. This kind of conflict exists in the case of several parameters describing the same entity on a service, and of a single parameter describing an entity on another WS. For instance a WS may describe the gender of a word as an independent parameter. Another service can describe a word gender in the “TYPE” parameter, separating the type from the gender with a special character like a coma. Thus parameters need to be merged in order to match those of the next WS.

3.2.3 The syntactic level.

Whereas in traditional data exchange, syntax conflicts usually happen between proprietary data storage methods, in the context of WS, the standard use of XML as an independent encoding language hides all conflicts that could happen. This is one of the advantages of using standard WS, relying on HTTP, SOAP and XML. However, it is noticeable that WS can use non-XML syntax languages, like EDI[11].

3.3 Interface Mediation

Following the assertion from the LSDIS team [23], we consider that an agreement is made on a common ontology, and we use the OWL-S language (described in section 2.2.1) for semantic description of WS interfaces. Ontologies are published over the Internet and thus represent a widely distributed and freely available reference. In our architecture, WS providers joining a JXTA peergroup (defined in section 4.1) must provide the appropriate OWL-S mappings from their WS interfaces to the ontology defined in the peergroup. As a consequence, even if there are differences between WS interfaces, it is possible to map each of them to the common ontology, thus solving semantic and

[11] http://www.x12.org/

structural heterogeneities (Figure 3). We do not detail the case of several ontologies inside a same peergroup, but this could be thought as a future work. Additional information can be lost from a WS interface to another, without particular consequences, because this extra information is not described in the ontology. Concerning syntax, the use of the independent language XML brings a complete standardized way of describing structures. As long as the user of a WS can parse XML, it is possible to communicate with a WS without any syntactic heterogeneities.

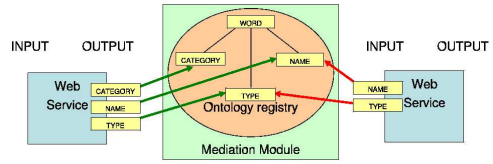


Figure 3. Mappings between input/output parameters and ontology

4 A Framework for a P2P platform

The use of P2P framework brings several advantages to client server architectures, mostly concerning the decentralization of resource advertisement repositories [25]. Our interest in making WS available within a P2P framework is to benefit from the advantages of P2P, without losing access to existing WS published in the standard SOAP/WSDL/UDDI architecture.

4.1 The JXTA Context

We rely on the open source JXTA project specification for several reasons. First, its Java implementation supports us with platform independence thanks to the Java virtual machine. Second, its standard Java implementation is relatively independent from the network topology and protocols, using either TCP/IP or HTTP as a transport layer, so that JXTA messages can pass through firewalls. Then, the use of the XML language in the transport layer provides application independence.

JXTA defines a peer as "any digital device connected to a network" [19]. Peers can belong to different JXTA peergroups[12], depending on the users' choice. Each peergroup gathers WS from a same domain of application, relying on ontologies in order to describe services and their interfaces. In practice, the classification into groups limits the range of search and selection. A peergroup can ask for membership authentication if required. JXTA also introduces the concept of advertisements, that describe available network resources like pipes, services (for instance WS), peers and other peergroups. Advertisements are published to participants of a peergroup in a decentralized manner [19]. They are the standard XML-based publication format defined in JXTA, though different from WSDL. There is a level of overlapping between WS and JXTA services at the advertising and discovery levels. The following details our framework for integrating WS into the JXTA model (Figure 4).

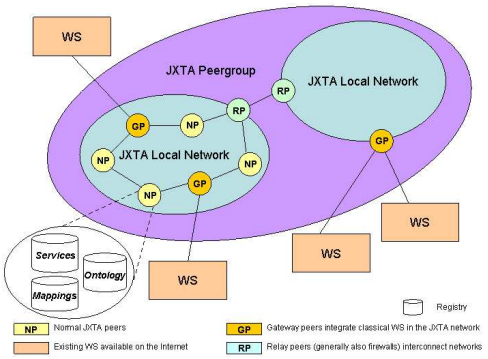


Figure 4. Integration of WS

4.2 Integration and Publication

In order to make existing WS available in the JXTA framework, we consider a peer that can access standard WS over the Internet (for instance through UDDI registries) and publish them as JXTA services (Figure 4). These peers, called gateway peers, provide a link (basically they act as gateways) between a JXTA network and the Internet. Their role is to describe any WS they provide in JXTA advertisements that are published on the P2P network. An advertisement contains both the WSDL and semantic (for instance OWL-S) descriptions of the WS as parameters, relying on the common ontology of the group. With this technique, all the peers get the semantic description and the WSDL of a WS without having to query any UDDI registry.

All JXTA peers have a local cache containing the advertisements sent over the network. Once published over the JXTA network as a service advertisement, a WS description is kept in every peer's local cache and considered as a normal resource. When a peer needs a service, it queries its local cache and selects the appropriate description, either a WS or a normal JXTA service description. Whether this is the description of an external WS or of a usual JXTA service is completely transparent to the user.

[12] Basically a group of peers

5 Implementation

5.1 Invocation Method

Depending on the strategy of centralization adopted within a JXTA group, completely decentralized, semi-centralized or centralized, gateway peers can act as proxies and provide external services to other peers like normal JXTA services, through pipe advertisements [19]. If a peer needs to use a published JXTA service, the gateway peer invokes the WS. In this case, the gateway peer acts as a proxy and forwards requests and answers in and out of the JXTA network. However, this approach is not recommendable. The advantages of decentralization provided with the P2P environment are lost if a centralized structure is built on top of it.

In our approach we propose a decentralized solution in which peers can all independently and directly invoke the WS endpoint outside of the JXTA network. For this purpose, we use a generic WS client engine adapted from an open source example [2] to our JXTA peers. This client engine extracts the necessary information (endpoint, protocol, parameters, expected answer)from the WSDL file provided as a parameter in the JXTA service advertisement, and builds the appropriate request at runtime. However, even if parsing a WSDL provides information about how to use a WS, there is a lack of information about semantics of its interface. Standard WSDL descriptions do not specify what meaning to give to data. There is a need for WS interface meta description, because when a peer wants to integrate a new WS in a composition, it needs to know how to compose it, technically how to map the results received from the WS output parameters to the input parameters of the next WS.

5.2 Application

In this section we overview the mediation process (Figure 5) using the example presented in section 3.1. A peer in the JXTA network composes available WS in order to form a composite WS returning a list of synonyms for a given French word. First, the peer looks into its local registry for descriptions of available services (WS or others) that can take part in the global service. A list is made of all the services that can be part of the global service. These services may independently be JXTA services or WS. Then, the JXTA peer maps the interfaces of selected WS to the ontology, using the OWL-S description from the advertisement. After storing interface mappings, the peer can invoke each WS in the correct order, or simultaneously if possible, and match their answers to the ontology. In our example both synonym dictionaries can be invoked in parallel, for instance with different threads, for a quicker response time. In a future work, the mediation code could be used for a simple structural mediation allowing answers aggregation, unfortunately with some data loss when a WS provide extra information.

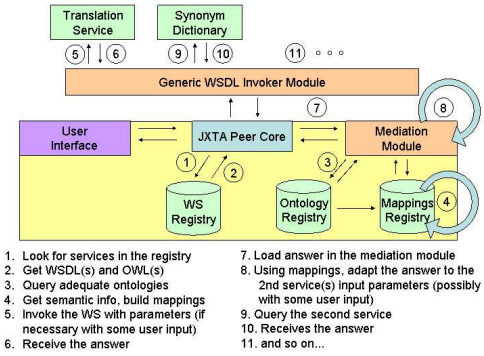


Figure 5. Mediation process

6 Conclusion and Future Work

In this paper, we review semantic and structural problems related to WS interface in a dynamic environment. As a proposal, we define an approach based on mediation components and domain ontologies. This approach consists OF defining a framework for WS integration into the JXTA architecture. We introduce the concept of gateway peer that establishes a link between standard WS and the JXTA model. As well, we rely on a generic WS client engine for WS runtime execution from their WSDL.

The use of ontologies constitutes a perspective to a larger subject encompassing ontology merge and translation, and in a broader area ontology middleware. As a future work, the idea of context-based information, introduced by Mamaar et al. with the OWL-C language, could help extending our framework to semantic discovery and selection.

References

[1] A. Ankolekar, M. Burstein, J. Hobbs, O. Lassila, D. Martin, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, T. Payne, and K. Sycara. Daml-s: Web service description for the semantic web, 2002.

[2] B. Behera and J. Winfield. Building Your Own SOAP Client and Reporting Tool. *Web Services Journal*, 2004. URL http://www.sys-con.com/webservices/article.cfm?id=674.

[3] B. Benatallah, M.-S. Hacd, C. Rey, and F. Toumani. Semantic Reasoning for Web Services Discovery. In *WWW2003 Workshop on E-Services and the Semantic Web, Budapest, Hungary*, May 20, 2003.

[4] B. Benatallah, B. Medjahed, A. Bouguettaya, A. Elmagarmid, and J. Beard. WebBIS: a system for building and managing Web-based virtual enterprises. In *Proceedings of the 1st workshop on Technologies for E-Services, in cooperation with VLDB2000, Cairo, Egypt*, September 2000.

[5] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *The Scientific American*, May 2001. URL http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21.

[6] T. Bultan, X. Fu, R. Hull, and J. Su. Conversation specification: a new approach to design and analysis of e-service composition. In *Proceedings of the twelfth international conference on World Wide Web*, pages 403–410. ACM Press, 2003.

[7] E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana. Web Services Description Language (WSDL) 1.1. *W3C, http://www.w3.org/*, March 2001.

[8] D. Fensel and C. Bussler. The web service modeling framework wsmf. Technical report, Vrije Universiteit Amsterdam, 2002.

[9] J. Gergic, J. Kleindienst, Y. Despotopoulos, J. Soldatos, G. Patikis, A. Anagnostou, and L. Polymenakos. An approach to lightweight deployment of Web Services. In *Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering*, pages 635–640. ACM Press, 2002.

[10] M. Gudgin, M. Hadley, J.-J. Moreau, and H. F. Nielsen. SOAP Candidate Recommendation Version 1.2. *W3C*, 2002.

[11] A. Heß, , and N. Kushmerick. Learning to attach semantic metadata to Web Services. In *International Conference on Web Services (ICWS'03)*, 2003.

[12] I. Horrocks. DAML+OIL: A reason-able web ontology language. In *Extending Database Technology*, pages 2–13, 2002.

[13] I. Horrocks, P.F.Patel-Schneider, and F. . van Harmelen. Reviewing the Design of DAML+OIL: An Ontology Language for the Semantic Web. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI 2002)*, pages 792–797, 2002.

[14] R. Hull, M. Benedikt, V. Christophides, and J. Su. E-services: a look behind the curtain. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 1–14. ACM Press, 2003.

[15] E. Leclercq, D. Benslimane, and K. Yétongnon. ISIS: A Semantic Mediation Model and an Agent Based Architecture for GIS Interoperability. In *Proceedings of the International Database Engineering and Applications Symposium, Montreal, Canada*, 1999.

[16] Z. Maamar and N. C. Narendra. Ontology-based Context Reconciliation in a Web Services Environment: From OWL-S to OWL-C. In *Proceedings of The 2nd International Workshop on Web Services and Agent-based Engineering (WSABE'2004) held in conjunction with The 3rd International Joint Conference on Autonomous Agents & Multi-Agent Systems (AAMAS'2004), New York, USA*, July 2004.

[17] A. Mädche and S. Staab. Services on the move - towards p2p-enabled semantic web services. In *Proceedings of the 10th International Conference on Information Technology and Travel & Tourism, ENTER 2003, Helsinki, Finland, 29th-31st January 2003*. Springer, 2003.

[18] S. Narayanan and S. A. McIlraith. Simulation, verification and automated composition of web services. In *Proceedings of the eleventh international conference on World Wide Web*, pages 77–88. ACM Press, 2002.

[19] S. Oaks, B. Traversat, and L. Gong. *JXTA in a Nutshell*. In a Nutshell. O'Reilly, September 2002.

[20] U. Ogbuji. Supercharging WSDL with RDF Managing structured Web service metadata. *IBM Developer Works*, 2000. http://www-106.ibm.com/developerworks/library/ws-rdf/?dwzone=ws.

[21] M. Paolucci, T. Kawamura, T. Payne, and K. Sycara. Importing the Semantic Web in UDDI. In *Proceedings of E-Services and the Semantic Web Workshop*, 2002.

[22] J. Schneider. Convergence of Peer and Web Services. *OpenP2P.com*, 07/20/2001. URL http://www.openp2p.com/pub/a/p2p/2001/07/20/convergence.html.

[23] K. Sivahanmugam, A. Sheth, J. Miller, K. Verma, R. Aggarwal, and P. Rajasekaaran. *Metadata and Semantics for Web Services and Processes*. Large Scale Distributed Information Systems (LSDIS) Lab, 2003.

[24] K. Sivashanmugam, K. Verma, A. Sheth, and J. Miller. Adding Semantics to Web Services Standards. In *International Conference on Web Services (ICWS'03)*, 2003.

[25] K. Verma, K. Sivashanmugam, A. Sheth, A. Patil, S. Oundhakar, and J. Miller. METEOR-S WSDI: A Scalable P2P Infrastructure of Registries for Semantic Publication and Discovery of Web Services. *To be published in the Journal of Information Technology and Management*, 2004.

[26] D. Wu, B. Parsia, E. Sirin, J. Hendler, and D. Nau. Automating DAML-S Web Services Composition Using SHOP2. In D. Fensel, K. Sycara, , and J. Mylopoulos, editors, *Proc. of the 2003 International Semantic Web Conference (ISWC 2003)*, pages 195–210, 2003. number 2870 in Lecture Notes in Computer Science.