Published paper
Kourtesis, Dimitrios and Paraskakis, Iraklis (2008) *Web Service Discovery in the FUSION Semantic Registry.* In: Business Information Systems : 11th International Conference, BIS 2008, Innsbruck, Austria, May 5-7, 2008. Proceedings. Lecture Notes in Business Information Processing, 7 (9). Springer Berlin Heidelberg , Germany, pp. 285-296.
http://dx.doi.org/10.1007/978-3-540-79396-0_25

Web Service Discovery in the FUSION Semantic Registry

Dimitrios Kourtesis and Iraklis Paraskakis

South East European Research Centre (SEERC),
Mitropoleos 17, 54624 Thessaloniki, Greece
{dkourtesis,iparaskakis}@seerc.org

Abstract. The UDDI specification was developed as an attempt to address the key challenge of effective Web service discovery and has become a widely adopted standard. However, the text-based indexing and search mechanism that UDDI registries offer does not suffice for expressing unambiguous and semantically rich representations of service capabilities, and cannot support the logic-based inference capacity required for facilitating automated service matchmaking. This paper provides an overview of the approach put forward in the FUSION project for overcoming this important limitation. Our solution combines SAWSDL-based service descriptions with service capability profiling based on OWL-DL, and automated matchmaking through DL reasoning in a semantically extended UDDI registry.

Keywords: Semantic Web Services, Web Service Discovery, Universal Description Discovery and Integration (UDDI), Semantic Annotations for WSDL (SAWSDL), Enterprise Interoperability.

1 Introduction

The Service Oriented Architecture (SOA) paradigm and its manifestation in the form of the Web services technology stack promise to become prime enablers for business agility in the modern enterprise by alleviating many of the barriers that stand on the path to achieving Enterprise Application Integration (EAI). Integrating a set of service-oriented business applications necessitates the assembly of services exposed by the individual business applications into new service compositions. This in turn requires discovering services that are suitable for performing each of the key tasks that a business process workflow comprises. Notably, in a fully SOA-enabled business application ecosystem with hundreds of deployed Web services, the task of searching and identifying the ones that are most appropriate for a certain type of use can become rather demanding.

This was the motivation behind the development of the Universal Description, Discovery and Integration (UDDI) specification [1] as a standardised way to catalogue and discover reusable services. UDDI registries however lack the means for supporting automated service discovery [2], [3], [4]. The reason is that indexing and retrieval in UDDI is not based on unambiguous, semantically rich representations of

Web service capabilities but on unstructured textual descriptions and categorisations that are retrievable through keyword-based search. Keyword-based annotation and search techniques cannot facilitate automated discovery since they do not provide any way of differentiating among (i) services that have identical naming but perform totally unrelated operations and (ii) services that have totally different names but offer equivalent functionality. To illustrate this problem through real-world examples, consider the case of two Web services that share "Address Validation" as their name but offer different functionality: the first one[1] validates postal addresses in the United States, while the second one[2] checks the validity of email addresses. Furthermore, consider the case of a service categorised by the name "UK Location"[3], able to check the validity of United Kingdom postal addresses, and another service categorised by the name "Global Address Verification"[4] which can still be of use for the exact same purpose, despite its apparently counter-intuitive name.

To overcome the problem of ambiguity that hinders automated service discovery we need to describe service characteristics in a formal, machine-understandable manner that is amenable to processing within semantically-enhanced UDDI registries. The aim of this paper is to present the approach adopted in project FUSION and the open source FUSION Semantic Registry[5] towards this direction, improving and elaborating on the preliminary work presented in [5]. FUSION is an EU-funded research project[6] aiming to promote efficient business process integration within and across enterprises, by offering a semantics-based solution to achieving interoperability among service-oriented business applications. The project aims at delivering a complete reference framework and a methodology for semantics-based Enterprise Application Integration (EAI), a reference implementation of the proposed framework, and a validation of the proposed approach through three pilot studies on intra- and inter-organisational integration. The introduction of semantics to Web service discovery is an essential requirement for realising the approach that FUSION puts forward, and encompasses: (i) describing service advertisements and service requests in a way that is formal, unambiguous and semantically precise, and (ii) realising a UDDI-based service registry that offers semantically-enhanced publication and discovery functions.

The rest of this paper is organised as follows. Section 2 introduces the requirements that FUSION puts forward for semantically describing service advertisements and service requests. Section 3 presents an overview of the FUSION Semantic Registry architecture, and provides a walkthrough on the core activities performed during service publication and service discovery. Section 4 analyses the matchmaking capabilities that the FUSION Semantic Registry supports, and its applicability for evaluating the relevance among a service advertisement and a service request at three distinct levels. Section 5 gives an overview and comparison of related work in this area, and section 6 concludes the paper with a small synopsis of the topics discussed.

---

[1] http://ws2.serviceobjects.net/av/AddressValidate.asmx?WSDL
[2] http://service.ecocoma.com/email/validate.asmx?WSDL
[3] http://www.webservicex.net/uklocation.asmx?WSDL
[4] http://ws.strikeiron.com/GlobalAddressVerification4?WSDL
[5] http://www.seerc.org/fusion/semanticregistry/
[6] http://www.fusion-strep.eu/

2 Describing Service Characteristics in FUSION

By using a semantic representation formalism to express the characteristics of Web services offered or needed, providers and requestors create definitions of service capabilities that are automatically processable through reasoning and logic-based inference. In turn, this can facilitate high-precision retrieval for services residing in a semantically-enhanced service registry, and offer a significant improvement over the capabilities of conventional UDDI registries. Evidently, the extent to which this can be achieved depends on the semantic representation formalism that is adopted for this purpose. The recent years have seen numerous Semantic Web Service frameworks being proposed and promoted for standardisation through W3C member submissions. The most prominent ones are OWL-S [6], WSMO [7], and the WSDL-S [8] specification that evolved into the W3C Recommendation of SAWSDL [9].

Although the FUSION reference framework does not prescribe the use of a specific Semantic Web Service framework, the reference implementation of the FUSION System that the Semantic Registry is part of builds on SAWSDL. In contrast to developing Web service descriptions at a high conceptual level and then linking these specifications to concrete Web service interfaces that are described in WSDL (as proposed in OWL-S and WSMO), the approach that SAWSDL puts forward is bottom-up: the WSDL documents themselves are to be enriched with annotations that capture machine processable semantics by pointing to concepts defined in externally maintained semantic models. The advantages of this approach are many-fold, but the most important one is that SAWSDL becomes agnostic to the knowledge representation formalism that one adopts. This allows service providers to annotate their services with concepts described in any modelling language, provided that these concepts are uniquely identifiable through URIs so that they can be referenced from within annotations. This promotes reusability for existing domain models and even allows SAWSDL to be used in conjunction with OWL-S or WSMO to combine the best of both worlds.

The semantic model that serves as the basis for creating, storing, and reasoning upon representations of service capabilities in the FUSION project is the FUSION Ontology [10]. Its multi-faceted structure reflects different types of concepts necessary for modelling a service: the data structures a service exchanges through messages (data semantics), the functionality categorisation of a service with regard to a taxonomy (classification semantics), and the behaviour it may expose within a complex and stateful process execution (behavioural semantics). The FUSION Ontology is encoded in OWL-DL, a Description Logics fragment of the W3C standard Web Ontology Language (OWL) that strikes a satisfactory balance between expressiveness and computational completeness [11] and facilitates decidable reasoning with the help of DL reasoning engines.

To represent the characteristics of a specific service advertisement or request in FUSION, one needs to create a Functional Profile, and define its key attributes in terms of references to the FUSION Ontology. A Functional Profile is expressed as a named OWL class that is attributed a set of three different OWL object properties:

i. `hasCategory`: associates a `FunctionalProfile` with a `TaxonomyEntity` concept from the service classification taxonomy that is part of the FUSION

Ontology, in order to represent the service's functionality categorisation. The cardinality of this property is exactly one.

ii. `hasInput`: associates a `FunctionalProfile` with an `InputDataSet` concept, in order to represent the set of data parameters that comprise the request message a service expects to receive and consume. The cardinality of this property is zero in the case of an *out-only* Message Exchange Pattern (MEP), or one, in the case of an *in-out* MEP.

iii. `hasOutput`: associates a `FunctionalProfile` with an `OutputDataSet` concept, in order to represent the set of data parameters that comprise the response message a service will produce if invoked. The cardinality of this property is zero in the case of an *in-only* MEP, or one, in the case of an *in-out* MEP.

Finally, each `InputDataSet` and `OutputDataSet` concept is associated with one or more `DataFacetEntity` concepts through a `hasDataParameter` object property, in order to represent the data parameters that comprise the message.

Depending on the perspective from which the Functional Profile is viewed, that of the provider or the requestor, we can differentiate among Advertisement Functional Profiles (AFPs) and Request Functional Profiles (RFPs). The first are created automatically by the FUSION Semantic registry at the time of service publication, while the latter are created by the service requestor at the time of discovery (or even at an earlier stage to be used as service request templates).

To allow for the construction of Advertisement Functional Profiles (AFPs), service providers need to augment the WSDL interfaces of their provided services with semantic annotations. The resulting SAWSDL interfaces must capture two elementary types of semantics: (i) the semantics of the data structures that a service exchanges through messages, and (ii) the semantics of a service's categorisation with respect to the functionality classification taxonomy. The semantics of a service's input and output data are captured by adding `modelReference` annotations to the appropriate `<xs:element>` entities under `<wsdl:types>`, while functionality categorisation semantics are captured with `modelReference` annotations on `<wsdl:portType>` entities.

3 An Overview of the FUSION Semantic Registry Architecture

There are many ways to realise a UDDI-based service registry that performs semantically-enhanced service matchmaking. A number of relevant attempts, each addressing a different set of requirements, are reviewed in section 5. The FUSION Semantic Registry architecture that is presented in this paper augments the purely syntactic search facilities that a UDDI registry can offer without requiring any modifications to the implementation of the UDDI server or the UDDI specification API, and this can be an important advantage compared to other approaches. We propose an architecture that positions a set of semantically-enabled modules externally to the UDDI server. These modules provide a specialised Web service API to the client, and are responsible for performing the necessary SAWSDL parsing, OWL ontology processing, and DL reasoning operations. Approaches based on this principle (i.e. relying on external components for specialised functionality while retaining the UDDI server implementation intact) have been also proposed in [4] and [12].

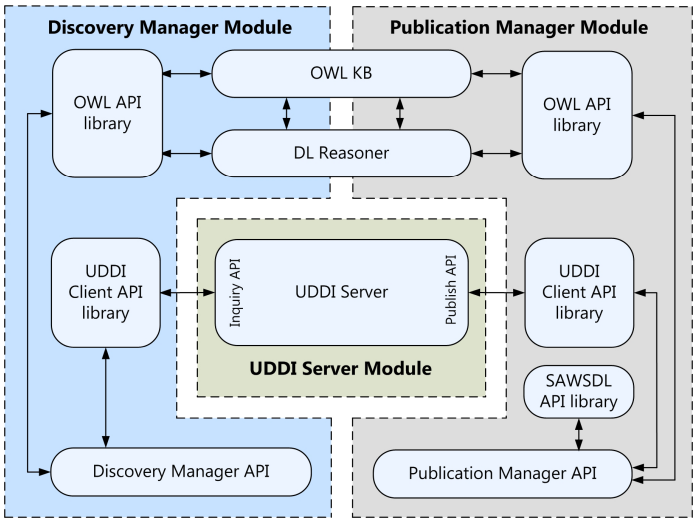

Fig. 1. FUSION Semantic Registry high-level architecture

As illustrated in Figure 1, the architecture that we propose comprises three core modules: (i) the UDDI Server Module, (ii) the Publication Manager Module, and (iii) the Discovery Manager Module. The UDDI server module is a typical server implementation of the UDDI v2 specification by OASIS [1] and a description of its functionality is beyond the scope of this paper. In the remaining of this section we focus on describing the functionality of the other core modules, and especially on the activities taking place during service publication and discovery.

3.1 Functionality of the Publication Manager Module

The Publication Manager Module provides an interface to the user for adding, removing, or updating descriptions of Web services (service advertisements), as well as adding, removing, or updating descriptions of service providers. A service provider can be a company, a business unit within an organisation, or even a specific business information system that offers some service on the network. The procedure of publishing a service advertisement comprises four phases:

i. Parsing an SAWSDL document to extract syntactic and semantic information
ii. Using the extracted semantic information to construct an Advertisement Functional Profile (AFP)
iii. Classifying the AFP in the registry's OWL Knowledge Base (KB)
iv. Mapping the extracted syntactic information and the derived semantic information to appropriate UDDI structures

The publication query that is used for initiating the publication process comprises four elements (i) the service provider ID (each service is associated with a specific service provider), (ii) a URL pointing to the SAWSDL document that contains the service description to be published (iii) an optional name for the service, and (iv) an optional free text description for the service.

The Publication Manager retrieves the SAWSDL document from the specified URL and extracts discovery-related information. Notably, the most valuable type of information to assist in discovery is not the syntactic characteristics of a service (e.g. its port and binding protocol information), but its defined categorisation and input/output messages, as already discussed in section 2. As depicted in Figure 1, this information is extracted with the help of an SAWSDL API library[7] that provides parsing and serialisation facilities. The syntactic and semantic characteristics that are extracted serve as input to a hybrid OWL-DL/UDDI indexing procedure.

Indexing begins by constructing an AFP and adding it to the registry's internal OWL Knowledge Base (KB) through the OWL API library[8] depicted in Figure 1. The Pellet DL reasoner[9] is subsequently used for performing an "eager" semantic classification of the new AFP against all known Request Functional Profiles (RFPs). The purpose of this classification procedure is to identify RFPs representing service requests that the newly added service advertisement can readily satisfy. We refer to this classification procedure as "eager" since it takes place at publication-time. In contrast, a "lazy" classification procedure would not have taken place before the actual need for matchmaking arises during discovery-time. This approach may be placing an overhead on the time required to complete the publication of a service advertisement, but it substantially reduces the time required to perform matchmaking at discovery-time, so it is considered particularly beneficial.

Three conditions must hold in order to claim that the new service advertisement can satisfy a service request: (i) the `InputDataSet` concept associated with the RFP must be subsumed by the `InputDataSet` of the AFP, (ii) the `OutputDataSet` of the RFP must subsume the `OutputDataSet` of the AFP, and (iii) the `TaxonomyEntity` concept associated with the RFP must subsume the `TaxonomyEntity` of the AFP. The interoperability-oriented rationale that these classification conditions reflect, and the way in which they collectively form a set of criteria for satisfactory matchmaking, is explained in section 4.

Finally, the Publication Manager maps the syntactic information extracted from the SAWSDL document and the semantic classification information derived by classifying the AFP onto appropriate UDDI data structures (`keyedReferences` to special-purpose `tModels`). Communication with the UDDI server module takes place through the UDDI Client API library, as illustrated in Figure 1. The mapping follows a well-defined methodology that is described in [13] and is beyond the scope of this paper to analyse. When the publication algorithm completes, a new semantic service advertisement has been created, registered with the UDDI registry, and is readily available for discovery.

3.2 Functionality of the Discovery Manager Module

The Discovery Manager Module provides interfaces for semantic matchmaking of a given service request against the published service advertisements, and for retrieving analytical information about records of advertisements and their providers.

---

[7] http://knoesis.wright.edu/opensource/sawsdl4j/
[8] http://owlapi.sourceforge.net/
[9] http://pellet.owldl.com/

The discovery query that initiates the semantic matchmaking process comprises two elements:

i. a URI pointing to some Request Functional Profile (RFP) that represents the characteristics of the Web service sought
ii. an optional system ID indicating the preferred service provider, i.e. the business information system that the service should originate from

The first step in the discovery procedure is to resolve the location of the RFP that is referenced by the provided URI. The RFP may be defined either within the FUSION Ontology that is shared by service providers and service requestors alike (i.e. be a shared RFP), or in some third-party ontology that imports and extends the FUSION Ontology (i.e. be a custom-built RFP). Depending on which of the two cases holds, the algorithm would follow a different discovery path:

i. If the RFP is defined within the FUSION Ontology, a syntactic, UDDI-compliant discovery query is generated and submitted directly to the UDDI server through the UDDI Client API library depicted in Figure 1.
ii. If the RFP is defined in a third-party ontology that is not shared with the service provider the Discovery Manager will load the ontology in which the RFP is defined to the DL Reasoner and compute the subsumption hierarchy.

Due to the shared ontology assumption that is valid in FUSION, the first case is the most typical type of discovery querying envisaged for the FUSION Semantic Registry, and is also the simplest and fastest type of matchmaking possible. Since the time-consuming process of concept classification has been already performed at publication-time, the computational complexity of discovery-time matchmaking for RFPs defined in a shared ontology is essentially as low as that of a conventional UDDI server.

The result of the discovery process is a list of advertisements complying with the matchmaking criteria captured by the RFP. If the optional system ID has been specified as part of the discovery query to indicate the preferred service provider, the registry uses it to filter-out services that are offered by systems other than the one specified. The ID is defined as an optional parameter in the discovery query, as it sometimes preferable to search for services that are offered anywhere within a service ecosystem, regardless of which business application exposes them.

4 Matchmaking Capabilities of the FUSION Semantic Registry

Due to the employed approach of OWL-DL-based service capability profiling and matchmaking, the FUSION Semantic Registry supports the evaluation of relevance among a service advertisement and a request at three distinct levels: (i) categorisation-level matching, (ii) message-level matching, and (iii) schema-level matching.

4.1 Categorisation-Level Matching

The end goal in this type of matchmaking is to determine if the categorisation value attributed to a service request is equivalent, more specific, or more generic than the

one specified in some service advertisement. As an example consider the case of a Request Functional Profile (RFP) classified under *Supply Chain Management* services, and some Advertisement Functional Profile (AFP) classified under *Freight Costing* services, a subcategory of *Transportation* services that is classified under *Supply Chain Management* services. As already discussed in the previous section, the `TaxonomyEntity` concept associated with an RFP must subsume the `TaxonomyEntity` of the AFP in order to have a match. In this example this obviously holds since the category of *Supply Chain Management* services with which the RFP is associated is more generic (subsumes) the *Freight Costing* services category of the AFP.

4.2 Message-Level Matching

The end goal in this type of matchmaking is to determine the degree to which a service can produce the set of output data that the requestor wants to obtain, and the degree to which the requestor can provide the set of input data that a service needs to receive when invoked. Positive matchmaking in this respect is essential for guaranteeing flawless communication and interoperability among a chain of composed services. By referring to sets of input and output data, instead of request and response messages, we intend to abstract from the differences among *complex* and *atomic* Web services. In the case of atomic, non-transactional Web service operations, the set of input data trivially corresponds to an operation's request message, while the set of output data corresponds to its response message. In the case of complex, transactional services involving the invocation of numerous Web service operations to fulfil one goal, the set of input data corresponds to the superset of all sets of input data exchanged as part of request messages for the operations involved, while the equivalent holds for output data.

As a result, the degree of match among the inputs or outputs of an AFP and an RFP would be determined by the degree to which their respective `InputDataSet` or `OutputDataSet` contain common elements. To provide a formal definition of degree of match we adopt the set-theoretic model from the work of [14] and [15]:

i. Exact Match: The advertisement consumes (for input-matching) or produces (for output-matching) the data that is exactly specified in the request
ii. Plugin Match: The advertisement consumes or produces all data specified in the request, but also consumes or produces some irrelevant data
iii. Subsumption Match: The advertisement consumes or produces only some of the data specified in the request, and no irrelevant data
iv. Intersection Match: The advertisement consumes or produces only some of the data specified in the request, but also consumes or produces irrelevant data
v. Non Match: The advertisement consumes or produces none of the data specified in the request

When checking for input message compatibility the cases of exact and subsumption match are the only ones that can be considered safe for interoperability and thus satisfactory for positive matchmaking. In the rest of the cases the advertised service is not guaranteed to receive all the input data it requires, and thus run-time errors could arise. Similarly, if we were checking for output message compatibility, the positive matchmaking cases are exact or plugin match. As a negative match

example, consider the case of an RFP representing a request for a shipment cost calculation service, having an `InputDataSet` that contains `Product` and `Customer` and an `OutputDataSet` that contains `ShipmentDetails`. Consider also an AFP with an `InputDataSet` that contains `Address`, `Product`, and `Customer`, and an `OutputDataSet` containing `ShipmentDetails` and `DigitalSignature`. Despite the fact that the `OutputDataSet` of the RFP subsumes the `OutputDataSet` of the AFP (i.e. the advertised service can offer more than what is being asked for), the `InputDataSet` concept of the RFP is not subsumed by the `InputDataSet` of the AFP (i.e. the advertised service asks for more than what can be provided).

4.3 Schema-Level Matching

The end goal in this type of matchmaking is to determine the degree to which the schema of some data parameter produced or consumed by an advertised service contains all the attributes specified in the corresponding schema of the request. When working under the assumption of a shared base ontology that can be specialised and customised for niche application domains through subclassing and applying quantification restrictions on properties, as in the case of the FUSION Ontology, the case may arise where different partners have chosen to extend a base ontology concept in different ways, thus creating potential interoperability problems. Figure 2 illustrates an example case in which the base concept of `Address` (depicted in the middle column) has been subclassed and specialised in two different ways, for modelling two different business applications.

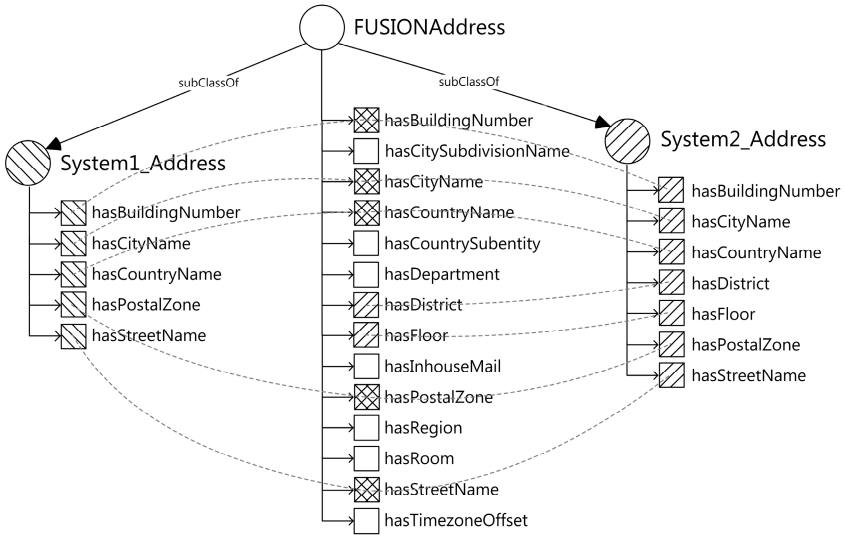

Fig. 2. Schema-level mismatch due to concept subclassing (excerpted from [5])

Although `System1_Address` and `System2_Address` are subclasses of the same concept, if they are used in the context of input and output data exchanged by the two systems, interoperability cannot be always guaranteed. The schema of `System2_Address` is

more specific than that of `System1_Address`, since the first specifies more attributes than the schema of the latter. In fact, if `System1` was to consume a service exposed by `System2`, and the service requested to be provided with address information as input, `System2` could consume all of the data included within `System1_Address`, but still require some additional data (`hasDistrict`, `hasFloor`) that would not be provided, thus leading to potential problems during process execution. As in message-level matching, the cases of exact or subsumption match could be considered satisfactory for positive matchmaking when checking for input compatibility, while the rest of the cases could not. When checking for output compatibility the cases considered satisfactory for positive matchmaking would be exact or plugin match.

5 Related Work

Recent years have seen an increasing interest on the use of semantics to represent service capabilities and on the introduction of semantic matchmaking functionality to UDDI registries, and numerous works could be considered relevant to ours. In this section we however discuss only related works that build on the established Semantic Web Service frameworks of OWL-S [6], WSMO [7], and WSDL-S [8].

In [2] and [16] the authors propose that discovery in UDDI registries should be realised through semantic matchmaking of service capability descriptions that are expressed as OWL-S Profiles and mapped onto UDDI structures. They propose the incorporation of a matchmaking engine inside the UDDI registry, thereby necessitating the modification of the UDDI server's interface and implementation. In a subsequent work [17] a revised mapping among OWL-S Profiles and the UDDI data model is proposed, and an improved version of the matchmaking algorithm from [16] is presented. Semantic classification and indexing are performed at publication-time rather than discovery-time and as pointed out by the authors in the paper the proposed solution is incomplete since discovery-time classification is not allowed.

In [18] the authors build on the approach proposed in [16] and present a method that improves the effectiveness of service discovery in UDDI based on a two-stage service discovery process, combining syntactic and semantic search. The expressiveness of the semantics that the proposed matchmaking algorithm employs are in the range of RDFS and OWL-Lite, and as a result the proposed solution cannot be used for matchmaking over highly expressive schema descriptions (e.g. with arbitrary cardinality restrictions on properties). Similarly to the approach by Paolucci et al, the solution proposed in [18] also necessitates some changes to the API and implementation of the UDDI server.

In [3], and later in [19], the authors present an approach for publishing semantically annotated WSDL descriptions based on a methodology for WSDL-S to UDDI mapping. Annotations are stored in UDDI and discovery is performed based on a semantic request template that captures abstract service characteristics. To perform matchmaking the described platform implements a semantic reasoner based on the Jena API. The reasoner supports semantic entailments for OWL-Lite but does not fully support OWL-DL, and therefore the proposed solution has some limitations as the one in [18].

A number of discovery engine implementations have been also developed in the context of the WSMX Working Group [20] for supporting the three different discovery approaches that are put forward in WSMO [15]: keyword-based discovery, lightweight semantic discovery (based on WSML-Rule and WSML-DL), and heavyweight semantic discovery (based on WSML-Flight). The specific works however do not offer themselves to direct comparison with our work or the other approaches discussed in this section, since they do not attempt to provide semantic enhancements to UDDI but rather stand as independent WSMX environment components and are not integrated with UDDI.

6 Conclusions

To promote interoperability among service-oriented business applications and efficient business process integration, the FUSION project promotes the introduction of semantics to Web service discovery in UDDI registries. In this paper we provided an overview of how UDDI, OWL-DL semantics, SAWSDL annotations and DL reasoning are employed within FUSION to enhance service discovery, we presented the FUSION Semantic Registry architecture and provided a walkthrough of the main activities performed during service publication and service discovery. Moreover, we analysed the matchmaking capabilities of the FUSION Semantic Registry and discussed its applicability in practical terms for evaluating the degree of match among service advertisements and service requests at three distinct levels: categorisation-level matching, message-level matching, and schema-level matching. To the best of our knowledge this the first attempt to combine SAWSDL-based service descriptions with OWL-DL based service capability profiling and automated matchmaking through DL reasoning in a semantically extended UDDI registry.

Acknowledgments. Research project FUSION (Business process fusion based on semantically-enabled service-oriented business applications) is funded by the European Commission's 6th Framework Programme for Research and Technology Development under contract FP6-IST-2004-170835 (http://www.fusion-strep.eu/).

References

1. Bellwood, T., Bryan, D., Draluk, V., Ehnebuske, D., Glover, T., Hately, A., Husband, Y.L., Karp, A., Kibakura, K., Kurt, C., Lancelle, J., Lee, S., MacRoibeaird, S., Manes, A.T., McKee, B., Munter, J., Nordan, T., Reeves, C., Rogers, D., Tomlinson, C., Tosun, C., von Riegen, C., Yendluri, P.: UDDI Version 2.04 API Specification, UDDI Committee Specification (July 2002)
2. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.: Semantic Matching of Web Service Capabilities. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, Springer, Heidelberg (2002)
3. Sivashanmugam, K., Verma, K., Sheth, A., Miller, J.: Adding Semantics to Web Services Standards. In: Proceedings of the 2003 International Conference on Web Services (ICWS 2003), Las Vegas, USA (June 2003)

4. Colgrave, J., Akkiraju, R., Goodwin, R.: External Matching in UDDI. In: Proceedings of the 2004 IEEE International Conference on Web Services (ICWS 2004), USA (July 2004)
5. Kourtesis, D., Paraskakis, I., Friesen, A., Gouvas, P., Bouras, A.: Web Service Discovery in a Semantically Extended UDDI Registry: the Case of FUSION. In: Camarinha-Matos, L., Afsarmanesh, H., Novais, P., Analide, C. (eds.) IFIP International Federation for Information Processing, Establishing the Foundation of Collaborative Networks, vol. 243, pp. 547–554. Springer, Boston (2007)
6. Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., Sycara, K.: OWL Web Ontology Language for Services (OWL-S). W3C Member Submission (November 22, 2004)
7. Bruijn, J.d., Bussler C., Domingue J., Fensel D., Hepp M., Keller U., Kifer M., Konig-Ries B., Kopecky J., Lara R., Lausen H., Oren E., Polleres A., Roman D., Scicluna J., Stollberg, M.: Web Service Modeling Ontology (WSMO). W3C Member Submission (June 3, 2005)
8. Akkiraju, R., Farrell, J., Miller, J., Nagarajan, M., Schmidt, M.T., Sheth, A., Verma, K.: Web Service Semantics (WSDL-S). W3C Member Submission (November 2005)
9. Farrell, J., Lausen, H. (eds.): Semantic Annotations for WSDL and XML Schema (SAWSDL). W3C Recommendation (August 2007)
10. Bouras, A., Gouvas, P., Mentzas, G.: ENIO: An Enterprise Application Integration Ontology. In: 1st International Workshop on Semantic Web Architectures For Enterprises, 18th International Conference on Database and Expert Systems Applications, Regensburg, Germany, September 3-7 (2007)
11. McGuinness, D.L., van Harmelen, F.: OWL Web Ontology Language Overview, W3C Recommendation (February 2004)
12. Pokraev, S., Koolwaaij, J., Wibbels, W.: Extending UDDI with Context Aware Features based on Semantic Service Descriptions. In: Proceedings of the 2003 International Conference on Web Services (ICWS 2003), Las Vegas, USA (June 2003)
13. FUSION project Deliverable D3.1 – Specifications of the Integration Mechanism (April 2007), http://www.fusion-strep.eu/
14. Li, L., Horrocks, I.: A Software Framework for Matchmaking Based on Semantic Web Technology. In: Proceedings of the 12th International World Wide Web Conference (WWW 2003), Budapest, Hungary (May 2003)
15. Keller, U., Lara, R., Polleres, A., Toma, I., Kifer, M., Fensel, D.: WSMO D5.1 – WSMO Web Service Discovery (v0.1). WSML Working Draft (November 2004)
16. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.: Importing the Semantic Web in UDDI. In: Proceedings of Web Services, E-Business and Semantic Web Workshop, Toronto, Canada, May 2002, pp. 225–236 (2002)
17. Srinivasan, N., Paolucci, M., Sycara, K.: Adding OWL-S to UDDI, Implementation and Throughput. In: Cardoso, J., Sheth, A.P. (eds.) SWSWPC 2004. LNCS, vol. 3387, Springer, Heidelberg (2005)
18. Akkiraju, R., Goodwin, R., Doshi, P., Roeder, S.: A method for semantically enhancing the service discovery capabilities of UDDI. In: Proceedings of the Workshop on Information Integration on the Web (IIWeb 2003), Acapulco, Mexico (August 2003)
19. Li, K., Verma, K., Mulye, R., Rabbani, R., Miller, J., Sheth, A.: Designing Semantic Web Processes: The WSDL-S Approach. In: Cardoso, J., Sheth, A. (eds.) Semantic Web Services, Processes and Applications, pp. 163–198. Springer, Heidelberg (2006)
20. WSMX (Web Service Modelling eXecution environment), http://www.wsmx.org/