Stefan Schulte, Melanie Siebenhaar, Julian Eckert, Ralf Steinmetz: *Query Languages for Semantic Web Services*. In: Fähnrich, Klaus-Peter and Franczyk, Bogdan: Service Science - Neue Perspektiven für die Informatik - Proceedings der Informatik 2010, Lecture Notes in Informatics, vol. 176, no. 2, p. 109-114, Gesellschaft für Informatik, September 2010. ISBN 978-3-88579-27

Query Languages for Semantic Web Services

Stefan Schulte, Melanie Siebenhaar, Julian Eckert, Ralf Steinmetz

schulte@kom.tu-darmstadt.de

Abstract: Even though service discovery is one of the vital steps in Web service invocation, service requests are usually expressed by rather simple means. While syntax-based service descriptions are usually addressed using keyword-based queries, Semantic Web Services are queried using "perfect" service descriptions. Hence, a service requester is not able to control the outcome of the discovery process in-depth by specifying similarity ranges or to explicitly state which syntax- and semantic-based parts of a service description should be addressed.

In this paper, we present an analysis of current approaches to service request formulation and identify requirements a query formalism for Semantic Web Services should fulfill. Furthermore, we give a brief preview on SWS2QL, a SPARQL-based query language for Semantic Web Services.

1 Introduction

Today, Semantic Web Services (SWS) are a prominent field of research and have resulted in a number of different approaches and standards such as *OWL-S*, *WSMO*, or *SAWSDL*, i.e., formalisms which explicitly make use of semantics in different parts of a service description.

One of the primary application areas of SWS is service discovery, which is essentially affected by three aspects: (i) The ability of service providers to describe their services, (ii) the ability of requesters to describe their requirements towards services, and (iii) the effectiveness of the service matchmaker, i.e., an algorithm that takes into account a request and finds the best fitting services from a set of service offers. The discovery of services is one of the core success factors for the invocation of services and is hence deemed one of the grand challenges in Web service research [PvdH07].

Service discovery research focuses mostly on matchmaking [Klu08], and the integration of SWS into service registries (e.g., [KP08, PKPS02]), two areas which have been examined from the very beginning of SWS research. In contrast, surprisingly little effort has been put into the deployment of alternatives to keyword- and service description-based query formulation. In the following, we will examine different ways to express a service request and identify their individual shortcomings (cp. Section 2). Based on this analysis, we specify requirements a service query formalism should fulfill (cp. Section 3). The paper closes with a description of SWS2QL, a query language based on SPARQL we developed in order to meet these requirements, and a conclusion.

2 Query Formulation for Semantic Web Services

In the following, we will give a short overview of different ways to express requirements towards a service:

Keyword-based service requests are the typical query formulation for syntactic Web services; this includes wildcard- and table-based queries. Keyword-based matchmaking is based on standard Information Retrieval (IR) methods, e.g., term matching: The terms from the request are matched with keywords found in a service advertisement. Explicit semantic information is not regarded.

Service description-based requests are widely adopted in research approaches which primarily aim on the deployment of new matchmaking algorithms (e.g., [KKZ09, PP09]). The basic idea is that the service request is formulated as a model instance, e.g., a WSDL-based service description. This model instance is deemed to be a "perfect" service, i.e., the perfect answer to the service request. While this is a very rich request format which can incorporate both syntactic and semantic information as well as the relationships between the single service components, there are a couple of shortcomings elaborated on in the following.

Query language-based service requests make use of a structured query syntax, e.g., a query language like SQL or SPARQL. Query language-based service requests do not directly represent a service as a model instance. Instead, parts of a virtual service model instance are *described* in terms of a well-defined query language. Service models provide the information and relationships required for creating a query language-based request. Query-based service requests can refer to both syntactic and semantic information.

Today, Web service portals like `www.seekda.com` primarily rely on keyword-based search which is not surprising as the listed services are mostly syntactically described [LS08]. The same applies to state-of-the-art service registry standards like UDDI and ebXML. As a result of the shortcomings of keyword-based retrieval, several approaches have already integrated semantic technologies into service registries, primarily UDDI (e.g., [KP08, PKPS02]). The most common approaches are built on top of syntactic Web service descriptions, require different execution environments and do not make use of potentially already existing (syntax-based) query formats. Alternatively, it is often possible to describe a service model instance, i.e., to follow the service description-based approach. However, this approach possesses several drawbacks:

Integration of syntactic search: Service descriptions do not allow for the explicit definition of syntactic information. It might be the case that a matchmaking algorithm takes syntactic information into account (e.g., [KKZ09, PP09]). However, this cannot be directly controlled by the requester if using a service description-based query.

Missing range: A service description does not recognize the definition of ranges for complete services and single elements. Thus, it is not possible to define, e.g., that only

those services which *exactly* meet the request should be returned by a matchmaker. Again, a matchmaking algorithm could define that only certain ranges are included in the result set of a query. However, it is not possible to set according parameters in the service query if using the service description-based approach.

Incompatible query formats: If a service request is formulated in a certain Web service formalism, it is not possible to discover service descriptions following another Web service standard. This forces users to formulate different queries if a service registry is able to manage more than one Web service formalism.

Lack of knowledge: Service requesters are not necessarily familiar with one or more service standards. However, requesters might be familiar with a query formalism which is, e.g., already used in a particular infrastructure or domain.

Usability: Neither the fact that a customization or parameterization of queries has to be done at the matchmaker-level nor the need to formulate queries for different Web service standards in different formats is very user-friendly. It is quite likely that the usage of semantic information in service requests will only be accepted by users, if it is possible to standardize Web service queries.

3 Specification of Requirements

In order to search for services based on semantic information, the requester needs to be able to define semantics in an intuitive and, if possible, uniform way, i.e., the request formulation should be independent from the service formalism(s) supported by the registry. Based on the shortcomings of the keyword- and service description-based approaches highlighted in the last section, the following generic requirements towards a query language for SWS have been identified:

Combination of syntactic and semantic search: A query language should integrate semantic as well as syntactic information. Furthermore, it should be possible to combine syntactic and semantic information within a single query.

Reuse: If a query language is already common to potential users, the chance to get it accepted will be likely higher. If multiple registry standards have to be supported, an appropriate mapping of the global query language to the specific registry query syntax should be provided.

Ranges: It should be possible to define a threshold of similarity a service offer needs to fulfill in order to be included in the result set.

As it was mentioned above, some of these requirements are currently reckoned in matchmaking algorithms, e.g., the usage of syntactic data or the definition of ranges or similarity thresholds. However, the actual handling of syntactic data or ranges in matchmaking is usually not explicitly regarded but used as a complement for semantic-based matchmaking. Hence, the usage of syntactic data and thresholds is not visible to the service requester

and cannot be directly controlled. In fact, such a control would make it necessary to offer the requester different setting properties of a matchmaking algorithm, which need to be manipulated apart from the actual query. The definition of further search constraints within a query is a much more intuitive approach.

Apart from the more generic requirements mentioned above, there are some requirements, which aim on the independence from concrete technologies and need to be regarded, too:

Flexible handling of different matchmakers: It should be possible to "manually" choose a matchmaker. If this is not done, the query processor needs to recognize appropriate matchmakers based on the actual query content.

Uniform query syntax: The query syntax should be the same for all Web service standards addressed by the corresponding service registry.

Handling of different registry standards: A uniform query syntax should be applicable to different registry standards.

4 SWS2QL – A Query Language for Semantic Web Services Based on SPARQL

Taking the requirements defined above into account, we have conceptualized and implemented the *SWS Structured Query Language* (SWS2QL). SWS2QL enhances SPARQL by new features aiming at SWS discovery. This is done based on the requirements described in Section 3. In addition, an abstract data model for services has been created. The actual query language, SWS2QL, is an enhancement of SPARQL and refers to this abstract model. The major benefit of making use of an abstract service model is the applicability of SWS2QL to different service standards which share common features with the abstract model. In the work at hand, these features are inputs, outputs, operations, interfaces, preconditions, and effects, which make it possible to map OWL-S- and SAWSDL-based services to the model and therefore apply SWS2QL to these SWS formalisms. As SAWSDL does not define preconditions and effects, we make use of *WSMO-Lite* in order to add these constructs to SAWSDL [VKVF08].

SPARQL has been selected as foundation for SWS2QL for several reasons. First of all, it is the major query language of the Semantic Web [KBM08]. Second, within SPARQL, query statements are expressed in terms of triples made up from a subject, object, and predicate. Thus, SPARQL represents a very intuitive approach, since the query statements are very close to shortened natural language-based expressions. Several enhancements of SPARQL for specific purposes have already been proposed as presented by [BFL$^+$09], showing the flexibility and adaptability of SPARQL towards different application areas. Although SPARQL is intended to be executed against RDF-based data, it may also be applied as a query *description* language only, as it is done in SWS2QL.

SWS2QL is compatible with the existing SPARQL grammar, makes use of existing extension principles, and minimizes the amount and complexity of further rules. SWS2QL

allows a service requester to address different service abstraction levels, incorporate and parameterize matchmakers, define similarity thresholds, etc. Due to the abstract service model, SWS2QL may be easily transferred to different service description and registry standards (through a mapping).

5 Conclusion

If comparing the six requirements defined in Section 3 with the work presented in this section, we can make the following statements: First of all, SWS2QL allows to combine syntactic and semantic search. Second, SWS2QL reuses and enhances an already known query syntax. Third, ranges can be defined by the definition of thresholds. Fourth, different matchmakers may be applied using SWS2QL as long as they implement an according interface. Fifth, SWS2QL is not restricted to one particular SWS formalism as it makes use of an abstract service model. Last, SWS2QL might be integrated into different service registry standards – however, a first proof of concept implementation has been done using ebXML.

Finally, we want to compare SWS2QL with the most relevant related work: Even though there are some approaches to establish query languages for SWS, they lack particular attributes regarding the requirements defined in the last section. To start with, Iqbal et al. present the usage of SPARQL ASK and CONSTRUCT queries to retrieve SAWSDL-based services [ISPG08]. More precisely, user goals are specified in the form of SPARQL ASK queries, which have to be fulfilled by a matching service, and the SPARQL CONSTRUCT query form is used for the representation of a service result including its pre- and post-conditions. Basically, this approach focuses on the definition of pre- and postconditions and sophisticated functionalities as, e.g., similarity ranges are not regarded. Kiefer and Bernstein also make use of SPARQL queries as SWS query language [KB08]. In their solution, *iSPARQL*, the authors propose the integration of imprecise matching capabilities into SPARQL. As the approach aims on imprecise matching, matchmaking is conducted using IR-based similarity values, which are applied to OWL-S service attributes (e.g., name, description) or sets of concept names resulting from the unfolded input and output concepts of a given service. In general, the approach proposed by Pantazoglou and Tsalgatidou, *USQL*, is closest to the work at hand [PT09]. Even though USQL provides a rich query model which allows to define syntax- and semantic-based query statements, thresholds, and may be applied to different service registry standards, it is only applicable to the proprietary PS-WSDL format, which is a major drawback of this approach.

In summary, SWS2QL represents a unified query language for SWS, which is not bound to specific service description formalisms, registry standards, or matchmakers like other service retrieval approaches. Concerning the query facilities, a holistic analysis of the requirements for a unified querying language for SWS has been performed. In doing so, the resulting query language SWS2QL is capable to overcome the shortcomings of related service retrieval approaches, while utilizing and enhancing the major query format of the Semantic Web, namely SPARQL, to facilitate user acceptance.

References

[BFL+09] François Bry, Tim Furche, Benedikt Linse, Alexander Pohl, Antonius Weinzierl, and Olga Yestekhina. Four Lessons in Versatility or How Query Languages Adapt to the Web. In *Semantic Techniques for the Web, The REWERSE Perspective*, volume 5500 of *Lecture Notes in Computer Science*, chapter 2, pages 50–160. Springer, 2009.

[ISPG08] Kashif Iqbal, Marco Luca Sbodio, Vassilios Peristeras, and Giovanni Giuliani. Semantic Service Discovery using SAWSDL and SPARQL. In *Fourth International Conference on Semantics, Knowledge and Grid (SKG 2008)*, pages 205–212. IEEE Computer Society, 2008.

[KB08] Christoph Kiefer and Abraham Bernstein. The Creation and Evaluation of iSPARQL Strategies for Matchmaking. In *5th European Semantic Web Conference (ESWC 2008)*, volume 5021 of *Lecture Notes in Computer Science*, pages 463–477. Springer, 2008.

[KBM08] Vipul Kashyap, Christoph Bussler, and Matthew Moran. *The Semantic Web, Semantics for Data and Services on the Web*. Springer-Verlag, Berlin, Heidelberg, 2008.

[KKZ09] Matthias Klusch, Patrick Kapahnke, and Ingo Zinnikus. SAWSDL-MX2: A Machine-Learning Approach for Integrating Semantic Web Service Matchmaking Variants. In *IEEE 7th International Conference on Web Services (ICWS 2009)*, pages 335–342. IEEE Computer Society, 2009.

[Klu08] Matthias Klusch. Semantic Web Service Coordination. In *CASCOM: Intelligent Service Coordination in the Semantic Web*, chapter 4, pages 59–104. Birkhäuser Verlag, 2008.

[KP08] Dimitrios Kourtesis and Iraklis Paraskakis. Combining SAWSDL, OWL-DL and UDDI for Semantically Enhanced Web Service Discovery. In *5th European Semantic Web Conference (ESWC 2008)*, volume 5021 of *Lecture Notes in Computer Science*, pages 614–628. Springer, 2008.

[LS08] Holger Lausen and Nathalie Steinmetz. Survey of Current Means to Discover Web Services. Technical report, STI Innsbruck, Austria, August 2008.

[PKPS02] Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, and Katia P. Sycara. Importing the Semantic Web in UDDI. In *International Workshop on Web Services, E-Business, and the Semantic Web (WES 2002)*, volume 2512 of *Lecture Notes in Computer Science*, pages 225–236. Springer, 2002.

[PP09] Pierluigi Plebani and Barbara Pernici. URBE: Web Service Retrieval Based on Similarity Evaluation. *IEEE Transactions on Knowledge and Data Engineering*, 21(11):1629–1642, 2009.

[PT09] Michael Pantazoglou and Aphrodite Tsalgatidou. The Unified Service Query Language. Technical report, National and Kapodistrian University of Athens, Greece, 2009.

[PvdH07] Mike P. Papazoglou and Willem-Jan van den Heuvel. Service oriented architectures: approaches, technologies and research issues. *The VLDB Journal*, 16(3):389–415, 2007.

[VKVF08] Tomas Vitvar, Jacek Kopecký, Jana Viskova, and Dieter Fensel. WSMO-Lite Annotations for Web Services. In *5th European Semantic Web Conference (ESWC 2008)*, volume 5021 of *Lecture Notes in Computer Science*, pages 674–689. Springer, 2008.