Web Services Annotation and Reasoning

Peter Graubmann, Evelyn Pfeuffer, Mikhail Roshchin

Siemens AG, Corporate Technology, Software and Engineering
{peter.graubmann|evelyn.pfeuffer}@siemens.com,
roshchin@gmail.com

Abstract: The aim of this position statement is to describe our work to support Web service technology as contribution to SOA by providing semantically "rich" descriptions of Web services together with methods and techniques to handle and to manipulate them by introducing Semantic Web technologies and additional logical formalisms into the annotation process. The annotation process will thus be enriched by new features and techniques to solve inference tasks (for instance with respect to consistency checking), to allow automated reasoning about annotation contents, to support automated search and complex query design, and to realize information derivation and interpretation based on different foci.

1 Introduction

Today, Web services are gaining momentum, and with good reasons. The Internet and XML standardizations provide flexible enabling solutions for service interconnection, invocation, composition in heterogeneous and complex environment. As a result, Web services paradigm is used for an increasing number of business solutions.

In the early days of SOA and Web service technology, the approach how to specify and describe services was over simplified and over emphasized. Now, experience and analysis have shown that a key issue for any success is an appropriate description of service semantics describing, for instance, functional and non-functional properties, requirements, behaviour, runtime prerequisites, etc. (see [2]).

Parallel to the Web services paradigm, the idea of the Semantic Web evolved within the realm of the Internet: Semantic Web basically means to associate semantics with information on the Web which in turn can be used to develop intelligent applications that collect information from various independent Internet sources, and that process and integrate this information automatically despite the fact that the meaning of the information on different places is not aligned directly and mutually agreed upon in advance. The main key of the Semantic Web is to describe a specific domain by ontology; that mainly means to capture the domain knowledge and to characterize the domain specific relations and inference rules. The latest standard for defining an ontology – proposed by W3C – is the Web Ontology Language (OWL) based on XML-syntax and the description logics formalism, which is employed to describe the rules that are used, for instance, by different inference tasks. The Semantic Web ideas provide much better results – if it comes to comparing and relating information from different sources, than, for instance, a simple keyword search, which doesn't make sense at all when the amount of information is growing exponentially. Service providers and requestors should be able to discover, invoke, compose, and orchestrate Web services, by specifying desired properties and requirements, behaviour and business logics, guaranties, qualities etc. OWL-S (formerly DAML-S) is an ontology of services that tries to make these functionalities possible. OWL-S supports Web service providers with a core set of mark-up language constructs for describing the properties and capabilities of their Web services in unambiguous, computer-interpretable form.

It turns out that the Semantic Web approach and desired additional mechanisms for the annotation process are quite similar tasks: the appropriate approach is to solve the problem of annotation composition and cooperation by extending the annotation process with a logical formalism of knowledge representation, and ontologies as the main part of the application domain description. That will provide the possibilities for automated reasoning about Web service properties (usable for service analysis, consistency checks and the extracting of information and its re-representation in different forms); it will further support the Web service search, by providing inference mechanisms that map pre- and post-conditions (with respect to the existing specifications) and the semantics of Web service properties.

Nevertheless, OWL-S is not the proper solution to represent semantics for Web services (for instance, doesn't have enough expressivity for describing Quality of Services (QoS), and is not aligned with the existing WS standards). There are a lot of other approaches (like METEOR-S [5] and WSMF [6]) that propose different ways to describe Web services from different focuses. Most of these approaches are based on different knowledge representation formalisms. And questions about semantics, expressiveness and adequacy don't get the attention they deserve.

We focused on the fact that existing standards of Web service description (WSDL) and business process execution (BPEL4WS) should be extended by additional semantic annotations of Web services to allow automation techniques for service discovery, mediation, composition, execution, consistency checking, and to support WS complete lifecycle and dynamicity.

2 Approach

Our approach is based on the idea to provide support for automated Web service composition. We integrate three approaches:

- The annotation framework, which allows a comprehensive semantic description of Web services and their behavior (based on Semantic Web technology);
- The connector concept (based upon the MSC language respectively the UML 2.0 interactions) for the component/interface behavior description [1] and in consideration of the relevance of system family approaches in the future, variability description for interaction protocols [3];
- Logic-on-demand roadmap to distinguish different levels of semantics which are needed to describe the require details and to allow the adequate reasoning about it.

This integration is based upon our so called Service Description Reference Model (SDRM) for which both, semantic description patterns and inference mechanisms are defined, determining the respective expressiveness of the involved description formalisms in association with the required analytic depth of the inference engines.

Our annotation framework offers a template-based description of Web services, in particular also with respect to usually “difficult” properties like behaviour, cross cutting concerns, or non-functional requirements/properties. The introduction of Semantic Web technologies into the annotation process enhances it by mechanisms that allow a number of important new features. These will comprise:

- Systematization of the domain descriptions:
  - Elaboration of domain specific ontologies written in OWL by providing formally defined classes and role names for the notions and relations within the domain and the respective annotations;
  - Identifying hierarchical structures (definitions of classes through other classes and roles) and properties for roles, (e.g., symmetry, transitivity, etc.);
  - Explicitly definition of ontologies that facilitate the provision of annotation templates; thereby, a better structuring of the annotation contents can be achieved and the exact relation of cognate notions can be more easily expressed.
- Automated procedures in the annotation process:
  - Provision of automated reasoning about annotations for many purposes, in particular to derive desired service properties which are not explicitly specified in the annotations;
  - Support of Web service selection and their mapping onto chosen architectures; automated search, invocation and composition through annotations, that will not necessarily be restricted to one domain (due to the capability of semantics comparison);
  - Use and adaptation of already known semantic descriptions, that will be used in order to extract and re-arrange annotation contents in order to meet specific user “languages” (as, for instance, spoken by marketing people or developers).
- Logical formalism as adequate description logics:
  - Inclusion of additional information (as for instance from design documents in formal languages like MSC, UML) into the search process based upon the annotations;
  - Support for automated configuration of components;
  - Consistency checking of annotations;
  - Transformation of annotations across domains.
- Logic-on-demand approach:
  - Description logic formalisms (based on OWL syntax) allow to describe needed knowledge on appropriate and adequate expressiveness levels based upon relevant concepts; rules to define these concepts;
  - Modal logic formalisms (for instance, temporal logic) allow us to describe real-time bounds and probability constraints on classes, that were predefined on ontology level;
  - Rule-based formalisms (for instance, RuleML, F-Logic, FOL) give a proper way to describe implications, criteria, rules for concrete values of classes and their instances.

In our work, as it was said above, we define the Service Description Reference Model. SDRM encapsulates the basic concepts for service annotation with respect to compositionality. On the one hand, it is intended to unify company based description models. And on the other hand, it allows the mapping service descriptions from other companies / different domains in order to make them comparable and accessible for component selection. The SDRM comprises three parts:

1. The Semantic Model,
2. The Logics and Inference Mechanism,
3. The Ontologies.

The Semantic Model is built by a hierarchy of so-called Semantic Micro-Models (SMMs). Each SMM defines a cluster of related properties, qualities, behaviours, etc. The contents of an SMM is characterised both, by its (domain or business) logical coherence and by its capability to be expressed within the same logics for knowledge representation (e.g., Classical FOL, Description Logics, Modal Logics, Non-monotonic Logics, etc.) as well as by the possibility to reason about and to analyse it with the same inference mechanisms. Thus, the various SMMs allow making variable use of semantic techniques. The SMMs form a lattice with respect to a consistency relation that is the basis to combine the various description clusters for complex description tasks.

The logics and inference mechanism are the collection of the logics and the corresponding inference engines that are used by the various SMMs.

The SDRMs ontology is comprised of:

- The general ontologies (which may be formed by any suitable collection of publicly available ontologies),
- The company-specific ontologies,
- The specific ontologies for the SMMs.

From the SDRM, templates are derived. They eventually are provided within the annotation framework. These templates ensure a correct and adequate domain specific semantic description of Web services. For the composition processes within the domain (for instance, within the company or one of its subdivisions), “matchmaking” is based upon these templates and the directly corresponding mechanisms in the SDRM.

If services from outside the company for which the SDRM is defined have to be incorporated, a semantic mapping of the Web service description into the SDRM has to be performed. Thereby, general available ontologies as well as publicly accessible domain-specific ontologies are used. The mapping mechanism also recurs to semantic models of the respective Web service and the logic, used for its description.

From the modelling layers point of view, we distinguish between:

1. The meta-model level, which includes SDRM and ontologies (or other forms of model represents) from outside.
2. The model level, consisting of:
   - Abstract annotations represent a kind of middleware between meta-model level and annotations, so that someone can define rules for calculation concrete values of properties (simply concepts, defined on the model level) or variability features, to distinguish values for different purposes (for instance, user groups). The formalism for knowledge description and specification is not fixed. Here, service provider can describe real-time features of QoS, variability of costs etc.

- The annotations represent finite model of concrete services for a concrete environment with concrete pre- and post-conditions, concrete user groups etc. Annotations are ready to be compared with service requestor requirements.

The evaluation of our approach could be seen by the application of our concepts on fidelity, as one of QoS characteristics. Fidelity turns out to be a complex concept to describe just within syntactic standards. Even in OWL is not clear how to describe this concept. The specification for fidelity should include probability functions and implication features (e.g. dependencies on some other functional or non-functional properties). We can apply the annotation level from our approach to describe the concrete value of fidelity for concrete Web service. On the abstract annotation level, service provider can specify suitable rules known for this concrete Web service (e.g. dependency on the number of sources, where the number of sources depends on the response time required by user).

3 Conclusion

We expect that the realization of our approach would enrich Web service technology. It provides automation techniques like service discovery, mediation, composition, execution, consistency checking, due to logical formalisms used for semantic description of WS. Also, it allows supporting the complete lifecycle and dynamicity of Web services.

Of cause, in our work we concern different approved methodologies, such as Web service standards, description specifications and logical formalisms. But still, there are open key issues for research in fields of how to deal with different logical formalisms in one conception or semantic specification; what is the proper way to introduce inference mechanisms to the annotation framework; how to work with them on different representational formalism levels. Currently we work on these problems.

References

1. Peter Graubmann: *Describing interactions between MSC Components: the MSC Connetctors*, Computer Networks 42, 2003.
2. Semantic Web Services Initiative (SWSI), http://www.swsi.org/
3. Maria Viktoria Cangarle, Peter Graubmann, Stefan Wagner: *From Feature Models to Variation Representation in MSCs*, Bosch Editor, Second Groningem Workshop on Software Variability Management, Technical Report IWI Preprint 2004/7/01, Rijksuniversiteit Groningem, 2004.
4. OWL-S, http://www.daml.org/services/owl-s/1.0/
5. METEOR-S, http://lsdis.cs.uga.edu/Projects/METEOR-S/
6. WSMF, http://wsmf.org/
7. Jack Greenfield, Keith Short: *Software Factories*, Wiley Publishing, 2004.