An ontology for semantic middleware: extending DAML-S beyond web-services

Daniel Oberle[1], Marta Sabou[2], Debbie Richards[3], and Raphael Volz[1]

[1] Institute AIFB, University of Karlsruhe, Germany
lastname@aifb.uni-karlsruhe.de
[2] Department of Artificial Intelligence, VU Amsterdam, The Netherlands
marta@cs.vu.nl
[3] Computing Department, Macquarie University, Sydney, Australia
richards@ics.mq.edu.au

1 Introduction

Describing software entities using Semantic Web technology is a growing research area. Our work investigates the semantic description of software entities that provide an application programmer's interface (API) to allow reasoning with the descriptions and their interrelationships. We present an ontology for our semantic middleware, called Application Server for the Semantic Web [3], where it is used to facilitate implementation tasks and semantic discovery. Building on an emerging standard from the Semantic Web community, our work includes a number of extensions to DAML-S [1] which currently allows semantic description of a particular type of software entities, viz. web-services, in order to facilitate their automatic discovery and integration.

2 The Ontology

Although DAML-S serves as a good starting point for our ontology, the main difficulty was in the type of software entities to be described. While DAML-S describes web-services, our goal is to describe software modules' APIs along other properties. As a result some parts of DAML-S were not reusable. Figure 1 presents the main ontologies in DAML-S in comparison with the ontologies we have developed. The following discussion is organised using our design principles.

1. Semantic vs. Syntactic descriptions We have adopted the separation between semantic and syntactic descriptions in order to achieve a flexible mapping. A number of our ontologies allow semantic description and others are used for syntactic descriptions. A mapping exists between the description of both aspects. However, given the different type of entities we want to describe, we modified some of the DAML-S ontologies as follows:

– we have kept the DAML-S *Profile* ontology for specifying semantic information about the described modules and extended it with a few concepts for describing APIs at the conceptual level which are grouped in a small ontology called *API Description*.

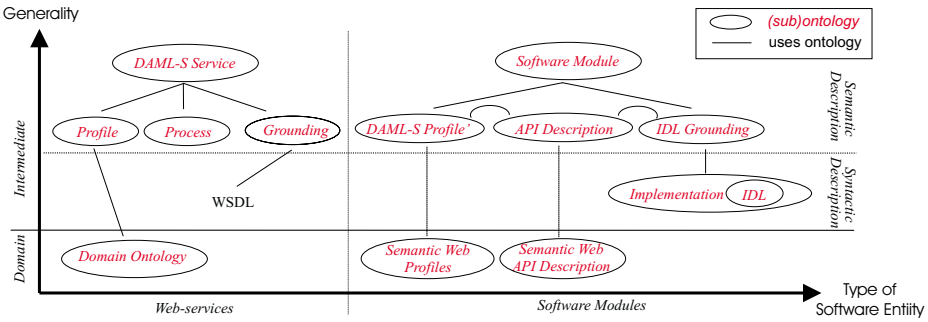

Fig. 1. Ontology overview

- we did not use the *Process* ontology since we are not interested in the internal working of the modules.
- we formalized a subset of *IDL* terms (Interface Description Language) and use them to describe the syntactic aspects of APIs in an *Implementation* ontology.
- as a consequence of the changes above, we could not reuse the existing DAML-S *Grounding* and wrote an *IDL Grounding* ontology.

2. Generic vs. Domain knowledge Our core ontology allows specifying semantic and syntactic knowledge about APIs in a generic way facilitating its combination with domain knowledge. For our semantic middleware we have built two domain ontologies in the area of the Semantic Web. The *Semantic Web Profiles* ontology specifies the type of existent Semantic Web software modules at a very coarse level. *Semantic Web API Description* describes the functionality of APIs at a more fine grained level. Our approach can be described in terms of the ONIONS [2] ontology development methodology which advises grouping knowledge with different generality in generic, intermediate and domain ontologies.

3. Modularity Modularity enables easy reuse of specifications and extensibility of the ontology. We apply this principle by making an effort to centralize related content to a certain concept whose instance can be reused at description time. We decided to group together chunks of information that are most likely to be reused. Like shown in Figure 1 we have grouped this information in small ontologies which are used by others.

References

1. M. H. B. et al. DAML-S: Web service description for the Semantic Web. In *Proc. of ISWC 2002*, volume 2342 of *LNCS*, pages 348–363. Springer, 2002.
2. A. Gangemi, G. Steve, and F. Giacomelli. ONIONS: An ontological methodology for taxonomic knowledge integration. In *Proc. of ECAI-96*, 1996.
3. D. Oberle, R. Volz, B. Motik, and S. Staab. *An extensible open software environment.* International Handbooks on Information Systems. Springer, 2003.