Modeling Services with Protégé

Daniel Elenius
Department of Computer and Information Science
Linköping University
581 83 Linköping, Sweden
daele@ida.liu.se

1 Introduction

For some years now, the Semantic Web [1] has been an active research field, promising to provide structure to the multitude of information on the web. Noy, et al. have shown how Protégé is an ideal tool for creating semantic web content in languages such as RDFS and DAML+OIL [2]. These ontology languages have now been superseded by the W3C Recommendation OWL[1] (Ontology Web Language), but the conclusion still holds true—due to its plugin architecture and flexible API, Protégé can easily adapt to changes in language specifications, while retaining its familiar frame-based conceptual view of ontologies. The support for OWL in Protégé is excellent, with the OWL Plugin, and tools such as OWLViz and OWLWizard[2].

In parallel with the evolution of the Semantic Web, web *services* have become a well-known concept. The aim is to connect service providers and service consumers on the web and making business processes more efficient by automating computer-to-computer interactions, using well-defined XML-based protocols such as WSDL[3] and SOAP[4].

In recent years, these two fields have spawned the *semantic web services* [3] research field. The idea is that if web *content* can be marked up with semantic information to improve searches on the web, then web *services* could be similarly marked-up in order to improve the intelligence and connectivity between service providers and service consumers on the web.

The focus of semantic web services research is currently on the OWL-S [4] ontology (which is in version 1.0 at the time of writing), which we give an overview of next.

[1] `http://www.w3.org/TR/owl-features/`

[2] These, and other Protégé plugins, are found at `http://protege.stanford.edu/plugins.html`

[3] `http://www.w3.org/TR/wsdl`

[4] `http://www.w3.org/TR/soap12-part1/`

2 An Overview of OWL-S

The OWL-S ontology consists of four main classes that specific services should instantiate. (Alternatively, service providers may create subclasses of the OWL-S classes and instantiate those instead).

- *Service*, with some basic concepts that tie the parts of an OWL-S service description together and holds a textual description of the service.

- *Profile*, which has properties used to describe what the service does—what it provides clients, and what it requires of them. More specifically, a service profile presents the inputs, outputs, preconditions and effects of a service. This information is used for matchmaking, i.e. to find an appropriate service based on its capabilities.

- *Process*, which has properties used to describe *how* the service works, i.e. what happens when the service is used. Services can be described as a collection of atomic or composite processes, which can be connected together in various ways, and the data and control flow can be specified. Service-seeking agents can use this information to perform a more in-depth analysis of the service, to compose several services to perform a task, to coordinate the activities of the services, and to monitor their execution.

- *Grounding*, with properties used to specify how the service is *activated*, including details on communication protocols, message formats, port numbers, etc. This 'abstract grounding' is usually tied to a 'concrete grounding' in the form of a WSDL interface description.

The figure below shows how the different parts of the OWL-S ontology fit together.

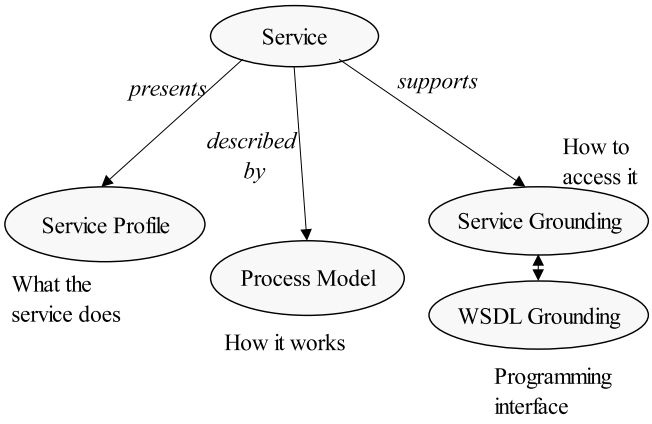

Figure 1: The main classes of the OWL-S ontology.

3 OWL-S Issues

Unfortunately, modeling services with OWL-S in Protégé is not as straightforward as the above description indicates. To understand why, we first need to discuss the different *sub-languages* of OWL—OWL Lite, OWL DL, and OWL Full[5]. These three sub-languages have increasing expressiveness, but also increasing complexity. OWL Lite is a subset of OWL DL, which is a subset of OWL Full, the most expressive of the three languages. OWL DL is so named for its correspondence to *Description Logics* [5], and provides maximum expressiveness while retaining computational completeness (all conclusions are guaranteed to be computable) and decidability (all computations will finish in finite time). While OWL DL is preferable over OWL Lite, OWL Full introduces unwanted complexity, as well as some constructs, such as classes of classes, that are controversial from an ontology engineering perspective. OWL DL has good tool support, both for ontology editing, and from inference engines such as Racer [6]. A problem with OWL-S is, as the reader may have suspected by now, that it makes use of OWL Full constructs. Protégé does not support OWL Full, only OWL DL, and OWL-S is therefore not immediately useable with Protégé, or most other editing tools or inference engines. However, work is underway (with OWL-S 1.1) to increase the tool compatibility of OWL-S and make it more (hopefully *completely*) OWL DL compliant. Due to space limitations, we will not go into further details on this here.

Another issue, discussed in [4], is that OWL does not give constructs that are sufficiently rich to express *data flow* in OWL-S. OWL-S therefore expands slightly upon OWL semantics and in some sense, it can be thought of as an extended language, requiring specialized reasoning methods in the most general case.

A third issue when it comes to modeling with OWL-S concepts is that it often becomes hard to get an overview of how the different parts connect to each other. For example, the same parameter (such as an input to a process) may be referenced in several places, and the control flow of composite processes may be of significant complexity.

4 Future Directions

We here discuss different ways in which semantic web services can evolve, and what impact these developments could have on Protégé.

4.1 Web Services Orchestration

A field that is closely related to semantic web services is industry initiatives for web service *orchestration* [7], with languages such as BPEL4WS[6]. Orchestration is the assembling of different web services that make up a business process, under one party's control. The idea is that the process logic, and the web services performing the work, should be separated in order to promote flexibility. BPEL4WS specifies a syntax for explicitly defining such compositions in an XML-based format. To acheive this, BPEL4WS must represent much of the same type of information that OWL-S deals with, such as the inputs and outputs of processes and subprocesses. However, BPEL4WS lacks the expressive semantics of OWL-S, and thus cannot support *reasoning* about services, and *automatic* composition of services. OWL-S, on the other hand, does not have support for BPEL4WS features such as persistence, transactions, or exception handling. For these reasons, combining OWL-S and BPEL4WS, or similar languages, is a promising direction for future research, as shown by [8].

4.2 Rules

A powerful extension to the semantic web in general, and semantic web services in particular, would be to layer *rules* on top of domain ontologies. Rule-based systems are not a new invention, but it is only recently that researchers have tried to integrate rules with ontologies for the semantic web. Encoding rules in ontologies in a standard format would enable interoper-

[5]The differences between these languages are further discussed in the OWL Language Reference, `http://www.w3.org/TR/owl-ref/`, Appendix E.

[6]`http://www-106.ibm.com/developerworks/library/ws-bpel/`

ability, reuse and exchange of rules, similar to the interoperability of domain ontologies written in OWL. It is also good knowledge engineering to make knowledge explicit—encoding the rules in ontologies rather than hard-coding them into rule-based systems. Most importantly, an ontology representation language for rules would also mean easier integration of rules and domain knowledge. Currently, specific translations have to be used, e.g. OWLJessKB[7] to load OWL ontologies into the Jess [9] rule-based system.

This development of rules has a direct impact on service modeling. OWL-S needs a rule language for at least two purposes: To express logical constraints and conditions on outputs and effects of processes; and to control how resources are consumed and produced by services. It is also likely that rules are useful more generally to express details about services that cannot be captured by static ontologies.

The DAML Rules project[8] are developing a rules language layered on OWL, called SWRL[9] (Semantic Web Rule Language, currently in version 0.5) which is a candidate for integration with OWL-S.

4.3 Problem-Solving Methods

Another research area that shows at least superficial resemblance to the issue of composing different services is Problem-Solving Methods (PSMs) [10]. This research focuses on reusable, domain-independent reasoning services. Domain independence is ensured by using a *method ontology* describing the properties of the method, and *mapping ontologies* that shows how the inputs and outputs of the PSM connect to entities in different domain ontologies. It would be interesting to explore this idea of mapping ontologies, to map the inputs, outputs, preconditions and effects of OWL-S services to different domain ontologies. The development of PSMs is tightly integrated with Protégé through the PSM Librarian widget.

5 Discussion and Conclusions

We have argued in this paper that semantic web services are an important paradigm, enabling automatic service discovery and interaction on the web. However, we have seen that service modeling, especially with new developments such as orchestration and rules, and given the complexity of the OWL-S ontology, is not entirely straightforward.

How does these developments affect Protégé? Or, conversely, how can Protégé have an impact on how service modeling is done? We beleive that Protégé should be *extended* (through the plugin architecture) to handle service modeling explicitly. This can be done in several complementing ways:

1. Services can be thought of as having an abstract (semantic) description, and a concrete (pragmatic) invocation interface. OWL-S, ontologies, and Protégé are well suited to handle the abstract part, whereas technologies like BPEL4WS (or at least a WSDL 'grounding', as mentioned above) are needed to make practical use of services. It would be highly useful to extend Protégé to handle also some aspects of the pragmatic part of services. This could include such things as an integrated WSDL editor, and an environment for actually executing composed services and seeing the results, while still being able to iteratively enhance the abstract conceptual model of the services.

2. The time it takes to get started with modeling an OWL-S service could be greatly reduced if there was an OWLWizard *Wizard* specifically engineered for OWL-S. This could create instances of the OWL-S service, profile, process and grounding classes based on streamlined user input.

3. It would be extremely useful to have a specialized graphical editing and visualization tool for OWL-S ontologies within the Protégé framework. While there already exist general-purpose visualization plugins, such as OWLViz, OWL-S has some specific requirements for such a tool. First, it should be possible to *edit* the service model, and not just visualize it. Using the standard OWL Classes tab to model a service is too limited for complex service models. An example of where OWL-S-specific extensions could be useful is when an input parameter is added to an OWL-S *process*. Often, the same parameter is also added to the *profile* and *grounding* of the same service. In an OWL-S editing environment, a dialog box could ask the user whether this should happen. Furthermore, the editor could support drag-and-drop composition of services, and a reasoner could be invoked to see which processes can be connected, and how.

All of the above extensions could be implemented in an OWL-S tab plugin for Protégé in an integrated way.

We envision that service modeling will become more important as more people come to realize the inherent potential in intelligent services on the web. As a leading modeling tool, Protégé will be expected to handle new

[7] `http://edge.cs.drexel.edu/assemblies/software/owljesskb/`

[8] `http://www.daml.org/rules/`

[9] `http://www.daml.org/2003/11/swrl/`

challenges in this field, and we have suggested ways that Protégé could be extended to do this.

Acknowledgements

The preparation of the manuscript was supported by Vinnova (grant no. 2002-00907) and by The Swedish Research Council (grant no. 621-2003-2991).

References

[1] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, vol. 284, pp. 34–44, May 2001.

[2] N. Noy, M. Sintek, S. Decker, M. Crubezy, R. Fergerson, and M. Musen, "Creating Semantic Web Contents with Protégé-2000," *IEEE Intelligent Systems*, vol. 16, March–April 2001.

[3] S. McIlraith and D. Martin, "Bringing Semantics to Web Services," *IEEE Intelligent Systems*, vol. 18, pp. 90–93, Jan–Feb 2003.

[4] The OWL Services Coalition, "OWL-S: Semantic Markup for Web Services," tech. rep. `http://www.daml.org/services/owl-s/1.0/owl-s.pdf`.

[5] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, eds., *The Description Logic Handbook - Theory, Implementation and Applications*. Cambridge: Cambridge University Press, January 2003.

[6] V. Haarslev and R. Möller, "RACER System Description," in *Proceedings of the International Joint Conference on Automated Reasoning, IJCAR'2001*, (Siena, Italy), June 2001.

[7] C. Peltz, "Web Services Orchestration and Choreography," *Computer*, vol. 36, pp. 46–52, October 2003.

[8] S. A. McIlraith and D. J. Mandell, "Adapting BPEL4WS for the Semantic Web: The Bottom-Up Approach to Web Service Interoperation," in *Proceedings of The SemanticWeb - ISWC 2003*, (Heidelberg), pp. 227–241, Springer-Verlag, October 2003.

[9] E. Friedman-Hill, *Jess in Action: Java Rule-Based Systems*. Greenwich, CT: Manning, 2003.

[10] M. Crubezy and M. Musen, *Handbook on Ontologies*, ch. Ontologies in Support of Problem Solving. International Handbooks on Information Systems, Springer-Verlag, January 2004.