An MDD annotation methodology for Semantic Enhanced Service Oriented Architectures

Lorenzo Pondrelli

Gruppo Formula, Via Mateotti 5, 40050 Villanova di Castenaso (BO), Italy
lorenzo.pondrelli@gruppoformula.it

Abstract. Service Oriented Architectures are increasingly being used to achieve interoperability among heterogeneous systems. SOAs help developers to integrate different architectures in order to reuse legacy systems and extend software boundaries. Semantics could improve SOA features adding a common understanding of the resources shared among different systems. Model Driven Development could be useful to produce a scalable and neutral methodology for guiding developers through software development and integration processes, employing reusable approaches and platform independency. In this context it is necessary to understand how we can use the MDD approach to enhance Service Oriented Architecture with semantics. This paper proposes to use a common visualization of ontologies and services, based on the industry-standard UML modelling language and its profiling system, to solve the resources annotation process in a SOA context. A first analysis of the main UML approaches for modelling ontologies is followed by the explanation of a general method for describing service interfaces and the resources exchanged among them.

1 Introduction

Semantic Oriented Architectures simplify the development of complex architectures, introducing the concepts of *black box components* and interface programming. *Black boxes* are units whose output is a specified function of the input, but for which the method of converting input to output is not necessarily specified [1], in other words pieces of software that can be used without knowledge of its inner workings, for which the user supplies the input assuming the output to be correct. Following this approach the most important thing in the developing and in the integration process is the definition of the interfaces of the software components and their description. If we want to ensure the interoperability among these interfaces we need to add not only a human understandable explanation but also a computer comprehensible common interpretation of the resources shared among the different systems. To achieve these results we can use two different approaches: using standards or using ontologies to share common knowledge.

2 An MDD approach for a semantic SOA

Ontologies could be used to build common knowledge that can be shared among different interfaces in order to exchange information between heterogeneous systems. In particular Domain Ontologies are formal organization of domain knowledge and in that way enable knowledge sharing between different knowledge-base applications [2], thus applicable to SOAs. In a SOA context we can use Domain Ontologies collections to annotate all resources, both interfaces and content schemas, in order to obtain shared descriptions of knowledge. There are software applications that allow the annotation of resources, in particular most of them permit the annotation of web pages (Ontomat [3], MnM [4], Cohse [5] and Smore [6]) or text documents (Trellis [7] and Melita [8]). Service Oriented Architectures are strictly related to the Web services technology and the associated standards such as SOAP [9], WSDL [10] and UDDI [11]. It is more difficult to find tools for the annotation of Web services and in general for the annotation of distributed application interfaces. One example is the METEOR-S Web Service Annotation Framework (MWFAS) [12] that produces WSDL-S descriptions of service interfaces. However for the purpose of this paper the key point of is the abstraction from particular technologies, such as WSDL and SOAP, generalizing from Web services technologies in order to find a more general solution. An MDD [13] approach should allowing platform independent development. In this context UML seems to be the best solution for proposing a complete developing framework for adding semantics to SOAs, in particular a profile for describing ontologies and a method for modeling *black box* interfaces which have been semantically enriched.

3 Using UML for modeling ontologies

The semantic languages, used to express ontologies, are not readable by humans because their representation syntaxes are built for machine understandability. From this perspective, one of the main open issues in this field is related to the visualization of ontologies for human readers. Most of the ontologies tools and editors, such as the most famous Protégé, use a tree arrangement to visualize ontologies following the structure of classes, slots, properties and instances. There are many proposals regarding the possibility to use UML, the industry-standard of modeling languages, for solving that issue. Some of the proposals consider the possibility to create a new MOF metalanguage at the same level of UML [14], [15] but most of them are related to UML profiles for ontologies [16], [17], [18]. The main difference is that the UML profiling system is not a first-class extension mechanism so it cannot be used to modify existing metamodels but only for adapting metamodels with constructs for a particular domain. It allows the extension of metamodels in order to adapt them for different purposes. The problem is that the ontology languages have many features that UML does not support. In this perspective could be useful to understand completely the relationships between the main ontology concepts and UML. Table 1 explains the most common similarities between them:

Table 1. Mapping between UML and the main ontology languages concepts

UML	Ontology concepts
Packages	Ontologies
Classes	Classes
Attributes, associations and classes	Properties
Navigable	Domain, range
Note	Comment
Multiplicity	Cardinality
Data types	Data types
Objects	Instances

The methods based on UML profiles allow the using of UML, so UML editors too, for building and managing ontologies. This fact means also that we can represent ontologies, abstracting from the different semantic languages, using XMI (now also Protégé contains a plug-in that includes a storage format compatible to the metadata standard MOF, obviously based on XMI [19]).

4 Semantic description of interfaces and their content

After the ontology modeling stage, the consequently step is the development of a methodology to semantically describe interfaces, such as Web services. Each interface needs two different levels of description. The first one regards the meaning of the interface itself and its operations. The second is related to the content of the objects exchanged among interfaces, in other words the meaning of the input and output parameters of each interface.
If we want to use UML as base of our annotation process, we need to profile both these two different kinds of resources:

- *Component Interfaces*, such as Web services.
- *Business Objects* to describe parameters exchanged among interfaces.

In [20], [21], [22], [23] there are proposals about UML profiles for services that can be used therefore to express C*omponent interfaces*. For *Business Objects* we can suppose to use an UML profile for XML schemas (as in [23]) that allows the description of semi-structured documents as proposed in [24], [25], [26], [27], [28], [29]. In this manner we can fully complete the description of all our resources, using UML and XMI, at a platform independent level.
The last step is the creation of a general technique for linking ontologies designed following a particular profile with other resources. Our proposal regards only the methodology and does not consider the choice of particular profiles. It is based on a double annotation:

– The *first phase annotation* is a visual annotation produced using UML dependency artifacts stereotyped with simples “semRef” labels or more specific stereotypes, between ontology elements and resource elements, such as service , attributes and operations. In Figure 1 there is an simple example of this kind of annotation using the Provost's service example proposed in [23] and a related ontology built following [16].

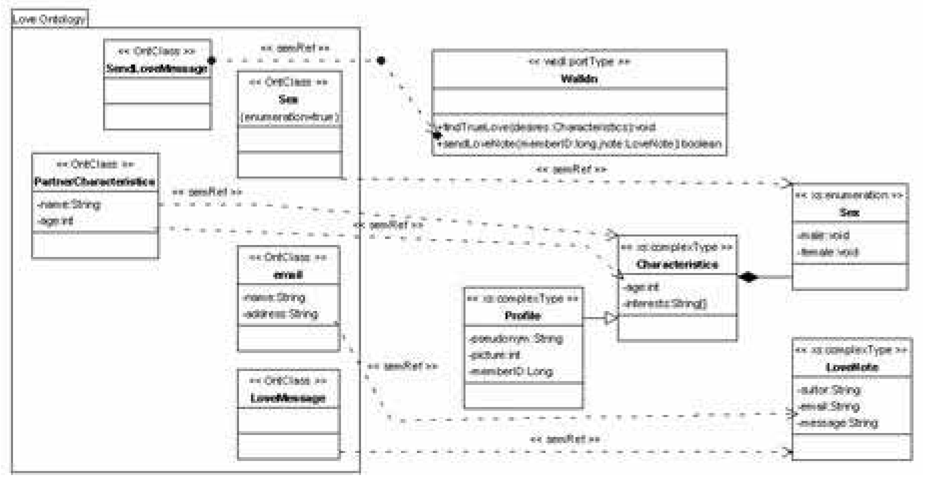

Fig. 1. *First phase annotation* example using "semRef" dependencies between Component Interface, Business Objects and ontology on the base of example proposed in [23].

– The *second phase annotation* produces a more detailed annotation performed using tagged values related to each resource elements. For instance, it is possible to add more exhaustive descriptions of the single parameters of the service operations or particular business information regarding services, such as Business Entities involved in the service lifecycle or Quality of Service information.
– Obviously, we can suppose to add also a method for including more human understandable information, using the UML notes system.

In this manner we ensure a complete platform independent description of the resources involved in SOA with the support of ontologies to guarantee interoperability.

5 Conclusions

Using our methodology it is possible to describe interfaces, such as Web services, and all the resources related, in a platform independent way using UML. The annotation process allows to add, at the same time, computer and human understandable metadata to each element of SOA resources, using common ontologies for sharing the information among different systems. In this manner we can suppose to use UML tools as ontology editors and annotators. XMI could become a neutral common language for describing elements related to ontologies and Service Oriented Architectures in order to utilize model transformation capabilities to get different Platform Specific Model for singular technologies.

References

1. Dictionary of technical terms for aerospace use, http://roland.lerc.nasa.gov/~dglover/dictionary/
2. Gruber, T. R., “A translation approach to portable ontology specifications”, Knowledge Acquisition, Vol. 5, No. 2, 1993.
3. OntoMat-Annotizer, http://annotation.semanticweb.org/ontomat/index.html
4. MnM, Ontology Driven Semi-Automatic and Automatic Support for Semantic Web, http://kmi.open.ac.uk/projects/akt/MnM/
5. Cohse, The Conceptual Open Hypermedia Project, http://cohse.semanticweb.org/
6. Smore, Semantic Markup, Ontology and RDF Editor, http://www.mindswap.org/2005/SMORE/
7. Trellis, I-Knowlegde-Capture, http://www.isi.edu/ikcap/trellis/
8. Melita, http://nlp.shef.ac.uk/melita/
9. Simple Object Access Protocol specifications. http://www.w3.org/TR/soap/
10. Web Service Description Language specifications, http://www.w3.org/TR/wsdl
11. Universal Description, Discovery and Integration specifications. http://www.uddi.org/specification.html
12. MWSAF. METEOR-S We Service Annotation Framework. http://lsdis.cs.uga.edu/Projects/METEOR-S/MWSAF/
13. Model Driven Architecture web site, http://www.omg.org/mda/
14. Kenneth Baclawski, Mieczyslaw M. Kokar, Jeffrey E. Smith, Evan Wallace1, Jerzy Letkowski, Manfred R. Koethe1 and Paul Kogut. UOL: Unified Ontology Language
15. Ontology Definition Metamodel.
16. D. Djuric, MDA-based Ontology Infrastructure, Computer Science and Information Systems (1) (2004) http://www.comsis.fon.bg.ac.yu/ComSISpdf/Volume01/Papers/DraganDjuric.pdf
17. K Baclawski, M. K. Kokar, P. Kogut, L. Hart, J. E. Smith, J. Letkowski, and P. Emery, Extending the Unified Modeling Language for ontology development, International Journal Software and Systems Modeling (SoSyM) 1(2) (2002) 142-156.
18. Cranefield Stephen, Purvis Martin. A UML profile and mapping for the generation of ontology-specific content languages. 2002
19. Protégé plug-in for XMI support. http://protege.stanford.edu/plugins/xmi/
20. R. Grønmo, D. Skogan, I. Solheim, J. Oldevik. Model-driven Web Services Development.
21. ACE-GIS. Adaptable and Composable E-commerce and Geographic Information Services. http://www.acegis.net/
22. IBM UML 2.0 Profile for software services. http://www-128.ibm.com/developerworks/rational/library/05/419_soa/#N10041
23. W. Provost. XML.com, 2003. Uml for Web services. http://webservices.xml.com/pub/a/ws/2003/08/05/uml.html?page=2
24. Martin Bernauer, Gerti Kappel, Gerhard Kramler. Representing XML Schema in UML - An UML Profile for XML Schema.
25. Nicholas Routledge, Andrew Goodchild, Linda Bird. XML Schema Profile Definition.
26. Nicholas Routledge, Andrew Goodchild, Linda Bird. UML and XML Schema. In 13th Australian Database Conference (ADC2002), pages 157–166. ACS, 2002.
27. D. Carlson. Modeling XML Applications with UML. Addison-Wesley, 2001.
28. R. Eckstein and S. Eckstein. XML und Datenmodellierung. dpunkt.verlag, 2004.
29. W. Provost. UML For W3C XML Schema Design. http://www.xml.com/lpt/a/2002/08/07/wxs_uml.html, August 2002.