Towards Semantic Service Specification and Discovery

Stanislav Pokraev[1], Roel J. Wieringa[2] and Maarten W.A. Steen[1]

[1] Telematica Instituut, P.O. Box 589, 7500 AN Enschede, The Netherlands
{Stanislav.Pokraev, Maarten.W.A. Steen}@telin.nl, http://www.telin.nl/
[2] Center for Telematics and Information Technology,
University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands
roelw@ewi.utwente.nl, http://www.ewi.utwente.nl/

Abstract. This document presents the research approach of the SEINE project. The goal of the research is to improve the existing methods for service specification and discovery by using ontologies. The product of the research will be an ontology-based method for semantically rich service specification and discovery and infrastructure that implements the proposed method.

1 Introduction

Service Oriented Architecture (SOA) is a one of the latest trends in software development. The essence of SOA is dynamic outsourcing of computational services. It is a modern-day variant of dynamic RPC across organizational boundaries through the web[3]. The dynamicity creates a need for service discovery and matching the capabilities of the requested and provided services. However, the current approaches for service discovery and matching offer low recall and low precision of the retrieved results. The reason is that they have to deal with informal and incomplete service descriptions, on the one hand, and informal and incomplete user requests, on the other hand. However, specifying all aspects of a service and a request formally puts an unrealistic burden on the service providers and service requestors and introduces enormous computational complexity for the underlying infrastructure. The goal of our research is therefore to develop methodological support for the service provider and the service requestor to specify and search for services. We will investigate how we can use current research in ontology specification to reach this goal.

In this paper we describe our research framework, which is the set of concepts we will use to describe and analyze services. Our framework is based on the GRAAL framework[2], which is itself based on the NYAM method[1] for specification of reactive systems.

2 The service discovery problem

It is not realistic to assume that different organizations will use the same terms to describe their services. For example, company A may decide to describe its money transfer service in terms of what are the input and output messages:

```
input:       {accountA, accountB, amount}
output:      {transactionId}
```

whereas company B may decide to describe the same service in terms state changes:

```
precondition:       {accountA.balance = x,
                     accountB.balance = y}
postcondition:      {accountA.balance = x – amount,
                     accountB.balance = y + amount}
```

Furthermore, different companies may use differing levels of abstraction to specify their services. For example, company A may just provide a description of its application interfaces, where company B may provide additional information about its business processes. Finally, the different companies may have different understanding about the meaning of the exchanged information. For example, company A's understanding on what "credit report" is may differ from the one that company B has, even if both companies share the understanding of the attributes of the "credit report".

3 The GRAAL framework

The GRAAL framework consists of a classification of a system properties or aspects. It takes as a starting point the distinction between *process* and *product* that results from this process, and focuses on the product aspects. The top-level distinction of product properties is between *external* and *internal* properties. External properties in turn are classified according to another well-accepted partition, namely *functional* properties and *quality* properties.

The basic aspect of the functionality is the system *service. Service* is an interaction that creates a desired, i.e. useful, effect in the environment, e.g. a delivered economic value, a satisfied need, the achievement of a goal, etc. *System communication* is interaction of the system with entities in its environment. Communication channels connect a collection of (sub)systems into a structure so they can interact whereas the *system behavior* is the ordering of these interactions over time. Finally, the *meaning* aspect consists of the semantics of the data exchanged during the service. The data exchanged are really messages received from and sent to entities in the environment. Their meaning is that they refer to a subject domain (often called Universe of Discourse). The services of the system are delivered because the messages sent out by the system have a certain effect, e.g. they provide information or change the status of the environment. Specification of service semantics therefore is specification of the meaning of messages in terms of a subject domain, and of the effects of messages in terms of the environment of the system.

Quality properties of a system indicate how valuable is the functionality of that system to the environment. In this group we can put properties such as efficiency, usability, reliability, etc.

The internal properties describe the composition of the product. In fact, this composition repeats the external properties at lower levels of aggregation.

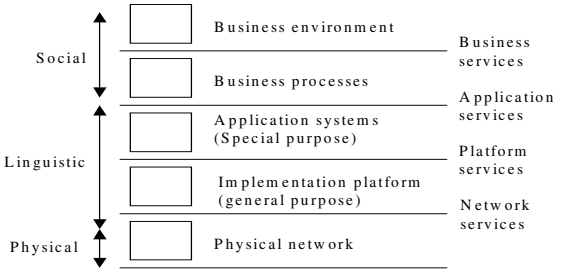

Figure 1

We round off our framework by structuring the context of a system into layers. We distinguish the physical, social and linguistic worlds and divide these worlds into layers as shown in figure 1. At each layer, we find systems, that have the aspects described previously in this section.

4 Research goal

The goal of this research is to improve the existing methods for service specification and discovery. This goal can be decomposed into smaller sub-goals:

1. **To define a methodology for describing services and service requests**. On the one hand, such a methodology should allow service providers to unambiguously describe both *functional* (service, behavior, communication and meaning) and *quality* (service area, service availability, etc.) properties of their services at *all system layers* in terms of what service *offers* to and what it *requires* from the service requestor. On the other hand, the method should allow the service requestors to express their needs, e.g. what they require from a service and what they are willing to provide in return.

How can we semantically describe the aspects of a service and a service request at all systems layers? The answer of this question will provide a methodological support for writing semantically rich service specifications and service requests. Furthermore, very often the service discovery task is a part of a more complex process, e.g. service outsourcing. Therefore, our methodology will provide guidelines for specifying service requests taking into account the constraints put by the existing business situation in a company.

2. **To define a method for matching of such service descriptions and requests**.

What is match between a service request and a service specification? Does the match between concepts and restrictions, used to express the service request and the service description, entail the match of the service request and the service description themselves? What is match between concepts, concept relations and restrictions on those relations?

5 Research approach

In our approach we argue that use of ontologies for specification of system aspects is advantageous. First, ontologies provide a *vocabulary* for modeling knowledge in a restricted domain. They are built by reaching a consensus within a community of interest and are key enabler for seamless knowledge interchange. Second, ontology languages are usually grounded with *formal semantics* such as model theory or description logic. This in turn enables *unambiguous definitions of compound concepts*. Based on these definitions it is possible to *infer new implicit information* from explicitly present information. Finally, the common vocabulary and precise mathematical specification of semantics open the way to *automatic information processing* since the information is not only understood by humans but also by machines.

In our approach we assume that domain-specific ontologies will be created by domain experts. Business parties will only refer to concepts from these ontologies when specifying their services and requests. In this fashion the formal specification of service aspects will become transparent for business parties. Moreover, we expect that in the future there will be tools that support semantic annotation of existing service descriptions (e.g. written in WSDL, BPEL or ebXML) which will make the additional effort for the service providers even less.

The combination of the layers presented and system aspects presented in section 2 is a basis for definition of a conceptual framework for service and request specification. The framework defines 4 main groups of concepts: *business ontology*, *application ontology*, *infrastructure ontology*, and a number of *domain specific ontologies* (figure 2). Some of the concepts in those ontologies are based on the concepts developed in ArchiMate project[4].

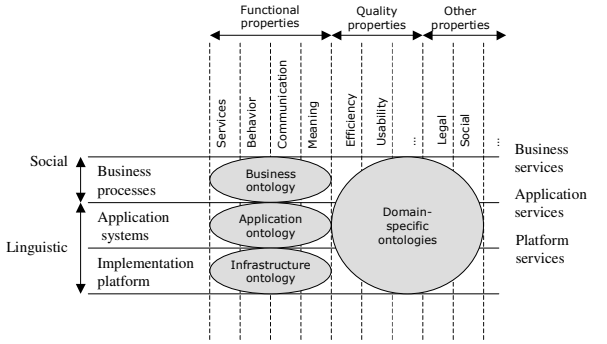

Figure 2

In our approach we propose to model services and service requests as instances of those ontologies. Next, the project aims at definition of a method for service discovery and matchmaking of such instances. The rest of this section describes the main concepts of the service specification ontologies.

Business ontology captures the information about business partners and the services that they require or provide. Example concepts from this ontology are *business actor, business object, business interface, business process, business function, business activity, business document, business product, business contract, business value*, etc.

Application ontology captures the information about software services that realize business processes within an organization. Example concepts from this ontology are

application component, application interface, application flow, application function, operation, message, etc.

Infrastructure ontology captures the information about hardware platforms and communication infrastructure needed to support the application services. Example concepts from this ontology are *infrastructure node, infrastructure interface, communication channel, infrastructure functionality*, etc.

Domain specific ontologies specify the quality properties of services. Examples of such ontologies are quality ratings (e.g. AAA diamond rating system), geographic ontologies, service category taxonomies (e.g. UNSPSC), device capability ontologies (e.g. CC/PP), etc.

6 Conclusions

In this paper we have outlined the problems of the existing service discovery methods and proposed an approach to improve them by the means of ontologies. The main thesis motivating this research is that existing approaches for service discovery offer low quality of the retrieved results because they have to deal with semantically poor service descriptions and requests. In addition, there is no methodology, that defines how service providers and requestors should describe the different aspects of their services and requests which results in descriptions that capture only limited aspects of a service and completely miss the service context. Finally, the lack of methodology creates a risk for mismatch of the abstraction levels used to describe a service. In this paper we proposed an ontology-based framework that defines a set of concepts to analyze and specify services. We would be very interested to get in touch with other members of the Interop NoE working on similar or closely related research questions. We also welcome feedback from peers on the validity and feasibility of the outlined research approach.

References

1. R.J. Wieringa. Design Methods for Reactive Systems: Yourdon, Statemate, and the UML. Morgan Kaufmann, 2003. http://www.mkp.com/dmrs
2. R.J. Wieringa, H.M. Blanken, M.M. Fokkinga, and P.W.P.J. Grefen. Aligning application architecture to the business context. In Conference on Advanced Information System Engineering (CAiSE 03), 2003. http://is.cs.utwente.nl/GRAAL/Wieringa_etal_caise03.pdf
3. G. Alonso, F. Casati, H. Kuno, V. Machiraju. Web Services. Springer Verlag, 2003.
4. Jonkers, H., Van Buuren, R. et al., "Towards a language for coherent enterprise architecture descriptions", in Steen, M.W.A and Bryant, B.R. (eds.), Proceedings 7th IEEE International Enterprise Distributed Object Computing Conference (EDOC2003), Brisbane, Australia, IEEE Computer Society, Sept. 2003, pp. 28-37.