

The Web Service Modeling Framework WSMF

Extended Abstract

D. Fensel¹ and C. Bussler²

¹ **Vrije Universiteit Amsterdam (VU)**

Faculty of Sciences, Division of Mathematics and Computer Science

De Boelelaan 1081a, 1081 HV Amsterdam, the Netherlands

Fax: +31-84-872 27 22, phone: +31-6-51850619

E-mail: dieter@cs.vu.nl

² **Oracle Corporation**

500 Oracle Parkway, Redwood Shores, 94065, CA, U. S. A.

Phone: +1-650-607-5684,

E-mail: chris.bussler@oracle.com

Abstract. Web Services will transform the web from a collection of information into a distributed device of computation. In order to employ their full potential, appropriate description means for web services need to be developed. For this purpose we define a full-fledged *Web Service Modeling Framework (WSMF)* that provides the appropriate conceptual model for developing and describing web services and their composition. Its philosophy is based on the following principle: *maximal de-coupling* complemented by *scalable mediation service*.

1 Introduction

The current web is mainly a collection of information but does not yet provide support in processing this information, i.e., in using the computer as a computational device. Recent efforts around UDDI, WSDL, and SOAP try to lift the web to a new level of service. Software programs can be accessed and executed via the web based on the idea of **web services**. A service can provide information, e.g. a weather forecast service, or it may have an effect in the real world, e.g. an online flight booking service. Web services can significantly increase the Web architecture's potential, by providing a way of automated program communication, discovery of services, etc. Therefore, they are in the centre of interests of various software developing companies. In a business environment this translates into the automatic cooperation between enterprises. An enterprise requiring a business interaction with another enterprise can automatically discover and select the appropriate optimal web services relying on selection policies. They can be invoked automatically and payment processes can be initiated. Any necessary mediation is applied based on data and process ontologies and the automatic translation of their concepts into each other. An example are supply chain relationships where a manufacturing enterprise of short-lived goods has to frequently seek suppliers as well as buyers dynamically. Instead of constantly searching for suppliers and buyers by employees the web service infrastructure does it automatically within the defined constraints.

Still, there need to be done more work before the web service infrastructure can make

this vision true. Current technology around UDDI, WSDL, and SOAP provide limited support in mechanizing service recognition, service configuration and combination (i.e., realizing complex workflows and business logics with web services), service comparison and automated negotiation. Therefore, there are proposals such as **WSFL** [Leymann, 2001] that develops a language for describing complex web services or **DAML-S** [Ankolenkar et al., 2001] that employees *semantic web* technology ([Fensel & Musen, 2001], and [Fensel et al., 2002]) for service description. The **Web Service Modeling Framework (WSMF)** follows this line of research. It is a full-fledged modeling framework for describing the various aspects related to web services. Fully enabled E-commerce based on workable web services requires a modeling framework that is centered around two complementary principles:

- Strong *de-coupling* of the various components that realize an E-commerce application.
- Strong *mediation* service enabling anybody to speak with everybody in a scalable manner.

These principles are rolled out in a number of specification elements and an architecture describing their relationships.

The contents of the paper is organized as follows. In Section 2, we provide a motivation for web services, an analysis of the state of the art of this technology, we identify eight layers as being necessary to achieve automatic web service discovery, selection, mediation and composition into complex services, and introduces the main principles of WSMF. Section 3, provides the architecture and the main modeling primitives of WSMF. Conclusions are provided in Section 4.

2 Web Services

The web is organized around URIs, HTML, and HTTP. URIs provide defined ids to refer to elements on the web, HTML provides a standardized way to describe document structures (allowing browsers to render information for the human reader), and HTTP defines a protocol to retrieve information from the web. Not surprisingly, web services require a similar infrastructure around UDDI, WSDL, and SOAP.

- **UDDI** provides a mechanism for clients to find web services. Using a UDDI interface, businesses can dynamically lookup as well as discover services provided by external business partners. A UDDI registry is similar to a CORBA trader, or it can be thought of as a DNS service for business applications.
- **WSDL** defines services as collections of network endpoints or *ports*. In WSDL the abstract definition of endpoints and messages is separated from their concrete network deployment or data format bindings. A port is defined by associating a network address with a binding; a collection of ports define a service.
- **SOAP** is a message layout specification that defines a uniform way of passing XML-encoded data. In also defines a way to bind to HTTP as the underlying communication protocol for passing. Instead of being document-based, automated B2B interaction requires integration of processes. SOAP is basically a technology to allow for “RPC *over the web*” providing a very simple one-way as well as

request/reply mechanism.

Many organizations had the insight that message definition and exchange are not sufficient to build an expressive web services infrastructure. In addition to UDDI, WSDL and SOAP standards for process definitions as well as exchange sequence definitions are proposed such as WSFL [Leymann, 2001], XLANG [Thatte, 2001], ebXML BPSS [Waldt & Drummond]. Still, there are important features missing in all of the mentioned frameworks. Very important is to reflect the *loose coupling* and *scalable mediation* of web services in an appropriate modeling framework. This requires mediators that map between different document structures and different business logics as well as the ability to express the difference between public visible workflows (public processes) and internal business logics of a complex web service (private processes). Therefore, we developed a full-fledged **Web Service Modeling Framework (WSMF)**. It provides a rich conceptual model for the development and the description of web services bringing this technology to its full potential. Fully enabled E-commerce based on workable web services requires a modeling framework that is centered around two complementary principles:

- Strong *de-coupling* of the various components that realize an E-commerce application. This de-coupling includes information hiding based on the difference of internal business intelligence and public message exchange protocol interface descriptions (see [Bussler, 2001a]). Coupling of processes can only be achieved via interfaces to keep the amount of interactions scalable.
- Strong *mediation* service enabling anybody to speak with everybody in a scalable manner. This mediation service includes the mediation of different terminologies [Fensel, 2001] as well as the mediation of different interaction styles [Bussler, 2001a].

The design of the WSMF is organized around these two principles.

3 The Web Service Modeling Framework WSMF

The WSMF consists of four main different elements: *ontologies* that provide the terminology used by other elements, *goal repositories* that define the problems that should be solved by web services; *web services* descriptions that define various aspects of a web service; and *mediators* which bypass interoperability problems.

3.1 Ontologies

Ontologies (cf. [Fensel, 2001]) are key enabling technology for the semantic web. They interweave human understanding of symbols with their machine processability. Ontologies were developed in Artificial Intelligence to facilitate knowledge sharing and re-use. Since the early nineties, Ontologies have become a popular research topic. They have been studied by several Artificial Intelligence research communities, including Knowledge Engineering, natural-language processing and knowledge representation. More recently, the concept of Ontology is also becoming widespread in fields, such as intelligent information integration, cooperative information systems, information retrieval, electronic commerce, and knowledge management. The reason ontologies are becoming so popular is largely due to what they promise: a shared and common

understanding of a domain that can be communicated between people and application systems.

3.2 Goal Repositories

The description of a *goal*¹ specifies objectives that a client may have in case he consults a web service. A goal specification consists of two elements:

- **Pre-conditions** describe what an web service expect for enabling it to provide its service.
- **Post-conditions** describe what a web service returns in response to its input.

Goal specifications should be kept separate from actual web service description because there is an *n2m* mapping between them, i.e., the same web service can serve different goals and obviously different (competing) web services can serve the same goal. For example, Amazon could be used to buy a book, however, in the same way it can be used as an information broker on bibliographic information about books.

3.3 Web Service

In the following we will elaborate on the various elements a web service may use to have a description with a certain level of complexity. First, a web service has a **name**, i.e., a unique identifier to refer to it. This requirement is elementary. Second, a web service fulfills a certain purpose, i.e., it should have a **goal reference**. This is the goal a service can achieve. Third, like goals, web service descriptions contain **pre conditions** and **post conditions** as introduced for goal descriptions. The pre condition is the condition that has to be true for the input in order for the web service being able to successfully execute. The post condition is the condition that holds once the complex service has been executed successfully, i.e., it defines constraints on its output. Forth, a web service description describes the structure of its **input data** and **output data**. Fifth, **error data** can be returned from the complex service through error ports at any time to indicate problems or error states. Sixth, a web service in turn may invoke other web services to provide its service. For each invoked web service a proxy called **invoked web service proxy** has to be declared. Seventh, a web service exposes input ports and output ports. Each connection between a complex service's input port and a invoked web service proxy's input port is a **data flow**. Eighth, data flow implicitly determines the execution sequence of invoked web service proxies as well as the steps within one complex service implementation. However, no all necessary execution sequences can be handled by data flow. For example, if data flow allows two invoked web services to be executed in parallel and they should be executed in a sequence, data flow is not capable of expressing this directly. The only way would be to have an artificial data flow between the two invoked web services. While this is possible, it is not good modeling practice at all. Instead, a **control flow** sequence should be introduced between the two invoked web services that defines the correct execution sequence. Ninth, web services may require **exception handling**. Tenth, after an invoked web service finally failed the invoking web service is in error state. A strategy of **compensation** for failed invoked web service is required. Eleventh, web services need description related to the **message**

1. [Fensel et al., to appear]

exchange protocol. Networks can be reliable as well as unreliable. Twelfth, there are important **non functional properties** that characterizes a web service.

3.4 Mediator

Adapters are of general importance for component-based software development. [Gamma et al., 1995] introduces an adapter pattern in his textbook on design patterns for object-oriented system development. Such adapters enable reusable descriptions of objects and make it possible to combine objects that differ in their syntactical input and output descriptions. [Fensel & Groenboom, 1999] introduced the concept of an *adapter* in architectural descriptions of knowledge-based systems to (1) *decouple* different element of this model, to (2) *encapsulate* these different elements and to (3) explicitly model their *interactions*. Work on software architectures describes systems in terms of components and *connectors* that establish the proper relationships between the former (cf. [Shaw & Garlan, 1996]). Here, the focus is to mediate between different interaction styles of components (which we call the business logic of a web service). Finally, work on heterogeneous and distributed information systems developed the concepts of wrappers and a mediator. Instead of assuming a global data schema, heterogeneous information and knowledge systems have a *mediator* [Wiederhold, 1992] that translates user queries into sub-queries on the different information sources and integrates the sub-answers.

For an open and flexible environment such as web-based computing, adapters are an essential means to cope with the inherent heterogeneity. This heterogeneity can wear many cloths:

- Mediation of **data structures**. A web service may provide an input for a second one, however, not in the format it is expecting.
- Mediation of **business logics**. Two web services provide complementary functionality and could be linked together in principle (one is a shopping agent and one is a provider of the searched goods), however, their interaction patterns do not fit.
- Mediation of **message exchange protocols**. SOAP over http is unreliable requiring trading partners have to implement transport level acknowledgments as well as time-out, retry, upper resent limits as well as duplicate detection in order to guarantee exactly once semantics.
- Mediation of dynamic **service invocation**. A web service may invoke other web services to provide its functionality. This can be done in a hard-wired manner, however, it can also be done more flexible by just referring to certain (sub-)goals. During execution other services can be invoked dynamically.

4 Conclusions

In this paper, we propose a modeling framework called **Web Service Modeling Framework (WSMF)**. Its main elements are: Ontologies, Service descriptions, elementary and complex Web Services. The aim of **WSMF** is to enable fully flexible *and* scalable E-commerce based on web services. We achieve this goal with an architecture that is based on two complementary principles: Strong *de-coupling* of the

various components that realize an E-commerce application and strong *mediation* service enabling anybody to speak with everybody in a scalable manner.

References

- [Ankolenkar et al., 2001]
A. Ankolenkar, M. Burstein, T. Cao Son, J. Hobbs, O. Lassila, D. Martin, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, T. Payne, K. Sycara, and H. Zeng: DAML-S: Semantic Markup For Web Services, <http://www.daml.org/services/daml-s/2001/10/daml-s.html>.
- [Bussler, 2001a]
C. Bussler: The Role of B2B Protocols in Inter-enterprise Process Execution. In *Proceedings of Workshop on Technologies for E-Services (TES 2001) (in cooperation with VLDB2001)*. Rome, Italy, September 2001.
- [Fensel, 2001]
D. Fensel: *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce*, Springer-Verlag, Berlin, 2001.
- [Fensel et al., 2002]
D. Fensel, J. Hendler, H. Lieberman, and W. Wahlster (eds.), *Semantic Web Technology*, MIT Press, Boston, to appear 2002.
- [Fensel et al., to appear]
D. Fensel, E. Motta, F. van Harmelen, V. R. Benjamins, M. Crubezy, S. Decker, M. Gaspari, R. Groenboom, W. Grosso, M. Musen, E. Plaza, G. Schreiber, R. Studer, and B. Wielinga: The Unified Problem-solving Method Development Language UPML, to appear in *Knowledge and Information Systems (KAIS): An International Journal*.
- [Fensel & Groenboom, 1999]
D. Fensel and R. Groenboom: A Software Architecture for Knowledge-Based Systems, *The Knowledge Engineering Review (KER)*, 14(3), 1999.
- [Fensel & Musen, 2001]
D. Fensel and M. Musen: Special Issue on Semantic Web Technology, *IEEE Intelligent Systems (IEEE IS)*, 16(2), 2001.
- [Gamma et al., 1995]
E. Gamma, R. Helm, R. Johnson, and J. Vlissides: *Design Patterns*, Addison-Wesley Pub., 1995.
- [Leymann, 2001]
F. Leymann: Web Service Flow Language (WSFL 1.0), May 2001. <http://www-4.ibm.com/software/solutions/webservices/pdf/WSFL.pdf>.
- [Shaw & Garlan, 1996]
M. Shaw and D. Garlan: *Software Architectures. Perspectives on an Emerging Discipline*, Prentice-Hall, 1996.
- [Thatte, 2001]
S. Thatte: XLANG: Web Services for Business Process Design, Microsoft Corporation, 2001. http://www.gotdotnet.com/team/xml_wsspecs/xlang-c/default.htm.
- [Waldt & Drummond]
D. Waldt and R. Drummond: EBXML: The Global Standard for Electronic Business, http://www.ebxml.org/presentations/global_standard.htm.
- [Wiederhold, 1992]
G Wiederhold: Mediators in the Architecture of Future Information Systems, *IEEE Computer*, 25(3):38—49, 1992.