SPECIFICATION OF A TOOL FOR MONITORING AND MANAGING A WEB SERVICES ARCHITECTURE

Youcef Baghdadi
Department of Computer Science, Sultan Qaboos University, Po Box 36 – PC 123, Al-Khoud, Oman

Keywords: Middleware, Web Services Architecture, Monitoring and Management Tool, Tool Architecture, Tool Specification.

Abstract: Enterprises willing to realize the service-oriented architecture with Web services, to gain advantages of an Internet and standard-based IT infrastructure, need to monitor and manage the deployed Web services architecture for an effective use, namely for flexible composition of business processes. The paper presents architecture and a specification of a tool for Web services monitoring and management. It mainly specifies the components of the architecture that are: (1) an information system that represents the properties of Web services architecture with different perspectives, namely their description, transport protocols, discovery, deployment platform, and the business processes composed out of them, and (2) the required monitoring and management artefacts built on top of the Web services architecture information system.

1 INTRODUCTION

Web services, principled by the service-oriented computing (SOC) paradigm to support low-cost composition of distributed applications (Huhns and Singh 2005; Papazoglou and Georgakopoulos, 2003), constitute a middleware that realizes an Internet and standard-based service-oriented architecture (SOA), which provides a methodology and a technology platform to re-architect the IT infrastructure. In fact, Web services: (i) have a machine-readable standard format (XML) of their specifications, and (ii) communicate through messaging protocols built on top of the Internet protocols (Curbera et al., 2003; Kreger, 2003).

Therefore, the Web services architecture, intended to be the main IT infrastructure, becomes an important asset of the enterprise. Accordingly, it needs to be monitored and managed for an effective realization and use, namely for composing flexible business processes.

The monitoring and management of a Web services architecture is an extension of the enterprise application management (Alonso et al. 2003). It enables a control over:

- The performance of each Web services with respect to an efficient use
- The performance of the platform, i.e. the server (Web server or application server), the Web services server, the Web services container, and the SOAP engine with respect to their workloads
- The performance of the internal and crossing business processes as well

This makes the Web services monitoring and management a complex task, especially when the number and types of Web services deployed grow without any control over them. Such a complexity requires tools that automate and support the monitoring and management efforts (Alonso et al., 2003; Casati et al., 2003; Papazoglou and van den Heuvel, 2005).

This paper presents architecture and a specification of a tool (with UML) for Web services monitoring and management. It mainly specifies two main components that are: (1) an information system representing the Web services architecture. It consists of three subsystems that represent the Web services, the platform, i.e. the servers, the container and the SOAP engine, and the business processes, and (2) a set of artefacts (subsystems built on top of the aforementioned information systems) to monitor and manage the Web services architecture, including Web services manager, business processes manager, servers manager, container manager, and SOAP engine.

The architecture of tool is meant to be flexible and scalable. Accordingly, it is made up of a set of cohesive UML packages. Each package contains the

Baghdadi Y. (2007).
SPECIFICATION OF A TOOL FOR MONITORING AND MANAGING A WEB SERVICES ARCHITECTURE.
In *Proceedings of the Ninth International Conference on Enterprise Information Systems - SAIC*, pages 51-56
DOI: 10.5220/0002408500510056

related elements of the aforementioned components of the Web services architecture. These are:

1. The Web services architecture information systems package that contains a set of class diagrams modelling the information related to the running Web services, the platform and the business processes.
2. The monitoring and management interface package contains the management use cases and the collaborations realizing them. The classes that participate in the collaborations are provided by the class diagrams of the Web services architecture information systems package.
3. The Web services interface package contains the modelling of the performance parameters of the Web services as running applications (within the Web services container), including their dependencies with each other and with other legacy systems such as databases and legacy applications.

2 WS ARCHITECTURE

There are many definitions given to Web service. Generally, Web services are software components provided by organizations, located on the Web, and accessible from any Web-connected application using a set of standard messaging protocols.

A definition close to this work considers a Web service as "a software application identified by a URI, whose interfaces and binding are capable of being defined, described, and discovered as XML artefacts. A Web service supports direct interactions with other software agents using XML-based messages exchanged via Internet-based protocols." (Austin et al., 2002).

In this work, Web services are considered as a new form of middleware based on industry standards such as XML, which is used as data format of the messages exchanged between the service and its clients. This middleware is implemented with a set of technology as shown in Figure 1, including SOAP, WSDL, and UDDI..

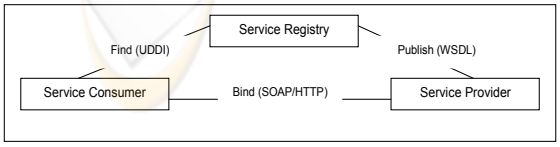

Figure 1: Web services architecture.

These XML-based technologies describe respectively the transport of the exchanged messages, the description, and the discovery of the service. Application can interpret a SOAP message using XML processing tools

2.1 Web Services Transport

The transport functional components define the data format, the protocol used to package the data into messages, and the protocol used to transfer the message.

SOAP which is based on Internet protocols is used to exchange XML messages communicating content: request, response, fault message and invoke operations offered by the services, where the SOAP data format is in XML, the encoding of data may be RPC-style or document-style, and the transfer may be any of the protocols such as HTTP, HTTPS, SMTP, POP3, IMAP, JMS, or other protocols.

2.2 Web Services Discovery

The discovery functional components provide a mechanism to register and find services. Web services architecture is implemented using UDDI, which is a mechanism for registering and finding services on the Web. A UDDI manages information about service providers and service types.

2.3 Web Services Description

The description of a Web service should be specified with a language that is readable by machine (Curbera, al., 2003). It concerns with: (a) the functional and non-functional requirements of the Web service, i.e. what functionality a Web service provides, (b) the communication style, i.e. how it communicates, and (c) its locations on the Web, i.e. where to find it. Accordingly, a description language such as WSDL expresses three distinct parts as shown in Table 1: (1) the abstract part expresses the functionality, (2) the concrete binding part expresses the communication, and (3) the implementation part expresses the location of the service. These three parts are related to each other by inclusion relationships.

2.3.1 Abstract Part

The abstract part expresses the interface of the service. It is structured as follows:

- Each Web service has one service interface (abstract part)
- Each service interface has one or more port types

- Each port type contains one or more abstract operations. An operation is a specification of the logic implemented by the service. The port type is equivalent to a component interface.
- An operation is a sequence of messages. An operation references a set of messages
- Messages are defined independently of the operations, so they can be used by other operations. A message may be: an input message that conveys the request content, output message that conveys the response content, or fault message that conveys errors exceptions. A message consists of a list of parameters (called parts), where a parameter is of a certain data type (e.g. String)

2.3.2 Concrete Binding

The concrete binding in a WSDL expresses the communication aspect and style, namely:

- The binding of the abstract part (or interface) to a concrete set of protocols
- The message structure: RPC-style or document-style
- The XML encoding data: encoding style or XML schema
- The protocol used to construct the envelop
- The header blocks to be included
- The transfer protocol: HTTP, FTP, or SMTP

The concrete part includes the abstract part

2.3.3 Implementation Part

The implementation part expresses the location of the Web services, namely:

- A collection of one or more related ports
- Each port implements a specific concrete binding of an abstract interface
- The port specifies the access point of service endpoint

A business may offer multiple access points to a particular service, each implementing a different binding.

The implementation part includes or imports the associated concrete binding.

2.4 Web Services Implementation

Web services are deployed within a Web services server, i.e. the run-time server. A Web service server can run standalone, a Web server, or within an application server provided by an environment such as J2EE or .NET.

A Web services server consists of a Web service container that manages the life cycle of the application implementing the services, and a SOAP processor that processes the exchanged messages.

The Web services container is responsible for:

- Managing the lifecycle of the application implementing the service
- Generating the WSDL which will be registered in the UDDI, where the client applications can find it and generate a client proxy. At run time, the client uses the proxy to construct and send SOAP message to the Web services

The SOAP processor is responsible for:

- Processing of incoming message
- Converting from XML into native PL data types
- Routing the request to the application that implements the service

Table 1: The three parts of Web service description and their relationships.

Part	Aspect	Content	Relation
Abstract Interface	What functionality	<types> <message> <portType> <operation>	//
Concrete Binding	How to communicate	<binding>	Includes Abstract Interface
Implementation	Where is it located	<service> <port>	Includes Concrete Binding

3 WS MANAGEMENT

The monitoring and management of a Web services architecture is an extension of the enterprise application management (Alonso et al. 2003). It enables a control over:

- The performance of each Web services with respect to an efficient use
- The performance of the platform, i.e. the server (Web server or application server), the Web services server, the Web services container, and the SOAP engine
- The configurability and the workloads of all the servers
- The performance of the internal and crossing business processes as well

This functionality of monitoring and management requires an information system representing:

1. The deployed Web services, including the three parts of their description
2. The platform, including the configuration and the workload of the Web services server, the Web services container, and the SOAP engine

3. The applications and business processes using the Web services

Therefore, a monitoring and management tool should consider different perspectives to come up with a unified management vision.

The management perspectives discussed here extend the business perspective, application perspective, and infrastructure perspective defined by Casati et al. in (Casati et al., 2003) to: (a) the deployed Web services, including the three parts of their description, (b) the platform, including the Web services server, the Web services container, and the SOAP engine, and (c) the applications and business process using the Web services. This extension comes up with the following elements that are categorized into: (C1) an information system representing the Web services architecture, and (C2) a set of artefacts (subsystems built on top of the different information systems) to monitor and manage the Web services architecture, where the built-in subsystems use the information provided by the information system.

The Web services architecture information system is made up of the following information subsystems:

1. The Web services information system that represents the information related to the deployed Web services, their dependencies with each other and with legacy systems such as databases and legacy applications
2. The platform information system, including information about the workloads and the configuration of the Web/Application server, the Web services server, the Web services container, and the SOAP engine
3. The business processes information system, including their respective flow and their composition in terms of Web services

The set of artefacts used for monitoring and managing the Web services architecture .are subsystems built on top of the previous information systems. These are:

1. A Web services management subsystem. It expresses the performance parameters of the Web services as running applications within the Web services container, including their dependencies with each other and with other legacy systems such as databases and legacy applications.
2. Four platform monitoring and management subsystems that express the performance parameters, the workloads and the configuration of (1) the Web services server, (2) the Web services container, (3) the SOAP engine, and (4) the business processes. It is worth noting that the four subsystems are depending on each other because the Web services container and the SOAP engine are depending on the Web services server, and the Web services server depends, in its turn, on the Web/Application server though it may be a standalone server. That is, the performances of the Web services and the business processes are depending not only on the performance of the Web services themselves, but also on the underlying platform.

4 TOOL SPECIFICATION

This section specifies the tool in terms of use cases specifying the monitoring and management functionality, the collaborations realizing them, and the class diagrams that participate to these collaborations. The use case and the collaborations model the built-in subsystems, whereas the class diagrams model the different information systems representing the Web services architecture. These use cases, collaborations and class diagrams are packaged into a Web services interface package, a management interface package, and a Web services architecture information system package.

4.1 Architecture Specification

The architecture of the tool is sketched out with UML in Figure 2, where:

1. The Web services architecture information systems package expresses the information systems related to the running Web services, the platform, and the business processes. Therefore, it contains three packages, where the business processes IS package depends on the Web services architecture IS package, which, in its turn, depends on the platform IS package.
2. The Web services interface package expresses the performance parameters of the Web services as running applications within the Web services container, including their dependencies with each other and with other legacy systems such as databases and legacy applications. This interface depends on the Web services architecture IS packages.
3. The management interface package expresses the management use cases and the collaborations realizing them. It contains four other packages that are: (i) the Web services server interface package dedicated to the management of the Web services server, (ii)

the Web services container interface package dedicated to the Web services container, (iii) the SOAP engine package dedicated to the management of the SOAP engine, and (iv) the business processes interface package dedicated to the composition and management of the business processes composed out of the deployed Web services. The four packages are depending on each other. The Web services container and the SOAP engine are depending on the Web services server, which, in its turn, depends on the Web/Application server.

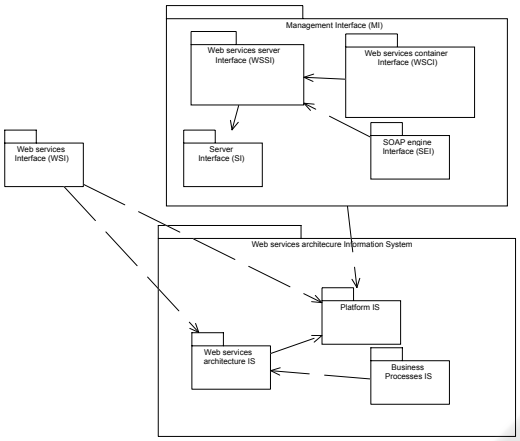

Figure 2: Architecture of the WS management tool.

4.2 WS Architecture IS

The information system representing the Web service architecture is specified with UML as shown in Figure 3. The Web services architecture information system represents the different perspectives, namely: (i) the deployed Web services, including the three parts of their description, (ii) the platform, including the Web services server, the Web services container, and the SOAP engine, and (iii) the applications and business process using the Web services.

4.3 Built-in Managers

The built-in subsystems (artefacts) are used to monitor and manage: (i) the deployed Web services, (ii) the business processes, the server; (iii) the Web/Application server, (iv) the Web services server, (v) the Web services container, and (vi) the SOAP engine. These managers ensure that the Web services architecture is flexible and configurable to make up for any change in the Web services, their platform workloads and configuration, and the business processes composed out of them.

4.3.1 Web Services Manager

The Web services manager handles information about the performance and the quality of services of the deployed Web services themselves (as running applications).

The required parameters may be categorized into configuration, workload, quality assurance, and statistics. The quality assurance parameters include the quality of services, security, dependability, cost, billing, and maintenance. The statistics are related to performance such resources consuming (e.g. CPU time, communication time, memory); and the use of the Web services such as the number of clients using the service by unit of time, the number of running copies of the service and so on.

4.3.2 Business Processes Manager

The business processes manager handles information such as: type of BP (internal, crossing), flow and the composition out of the Web services, flexibility, dependency, performance, and number of occurrences.

4.3.3 Web/Application Server Manager

The Web/Application server manager handles information about: performance of the server, number of services running simultaneously, load balancing, and workload.

4.3.4 Web Services Server

The Web services server manager handles information about: performance of the server, number of services running simultaneously, load balancing, and workload.

4.3.5 Web Services Container Manager

The Web services container manager handles information such as: performance, service life cycle, and number of generated WSDL.

4.3.6 SOAP Engine Manager

The SOAP engine manager handles information such as: performance, number of messages communicated, and number of proxies generated.

5 RELATED WORK

Although the Web services architecture is a critical asset for any enterprise re architecting its IT

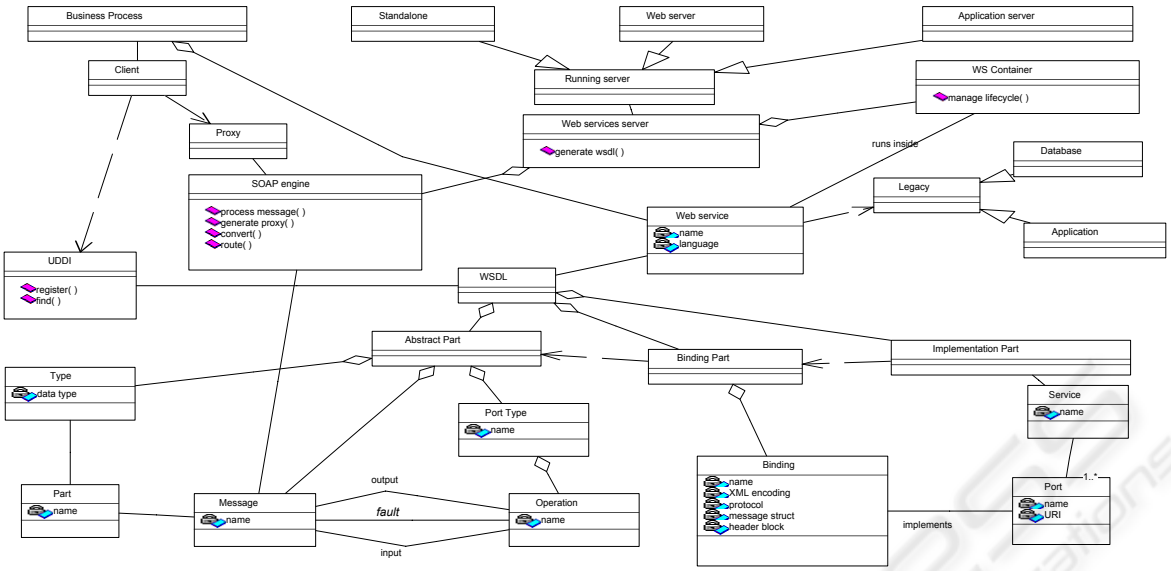

Figure 3: Class diagrams for the Web services architecture information system.

infrastructure with SOA, only few work concern with their management. The authors in (Alonso et al. 2003), in their synthesis work on Web services, consider the management a critical issue; but they only set the principles for the monitoring and management. In their work on the extended SOA, Papazoglou al. in (Papazoglou and Georgakopoulos, 2003; Papazoglou and van der Heuvel, 2005) have added a layer to SOA dealing with management perspectives. The authors in (Casati et al., 2003) have categorized the perspectives of Web services management into business, applications, and infrastructure perspectives.

The approach presented here is an extension of the aforementioned work. IT mainly aims at specifying the management tool including its architecture with well-known modelling language that is UML.

6 CONCLUSION

This work has specified a tool for Web services architecture monitoring and management. The specification has taken into account different perspectives of the Web services architecture. It is expressed with UML in order to be readily and promptly implemented.

The architecture of the tool mainly consists of three packages, where each package contains other related packages, having in mind the design decisions such as cohesion and coupling to make the tool flexible and scalable.

The information system representing the Web services architecture has been specified in term of class diagrams modelling al the properties.

This tool is readily extensible to capture an exhaustive list of monitoring and management properties.

REFERENCES

Alonso, G., Casati, F., Kuno, H. and Machiraju, V., 2003. *Web services: Concepts, Architecture and Applications*, Springer.

Austin, D., Barbir, A., Ferris and Garg, S., 2002. Web Services architecture requirements. *In W3C Working Group Draft 14.*

Casati, F. Shan, E., Dayal U. and Shan M., 2003. Business-oriented management of Web services. *Communications of the ACM, Vol. 46, No. 10, pp. 55-60.*

Curbera, F., Khalaf, R., Mukhi, N., Tai, S. and Weerawarana S., 2003. The Next step in Web Services. *Communications of the ACM, Vol. 46, No. 10, pp. 29-34.*

Huhns M. N. and Singh, M. P., 2005. Service-oriented computing: Key concepts and principles. *IEEE Internet Computing (January-February), pp. 75-81.*

Kreger H., 2003. Fulfilling the Web services Promise. *Communication of the ACM. Vol. 46, No. 6, pp. 29-34.*

Papazoglou M. P. and van den Heuvel W. J., 2005. Web service management: A survey. *IEEE Internet computing Nov-Dec, pp. 58-64.*

Papazoglou, M. P. and Georgakopoulos, D., 2003. Service-Oriented Computing. *Communications of the ACM Vol. 46, No. 10, pp. 25-28.*