Chapter 10
Web Service Specifications Relevant for Service Oriented Infrastructures

Eduardo Oliveros
Telefónica Investigación y Desarrollo, Spain

Jesús Movilla
Telefónica Investigación y Desarrollo, Spain

Andreas Menychtas
National Technical University of Athens, Greece

Roland Kuebert
University of Stuttgart, Germany

Michael Braitmaier
University of Stuttgart, Germany

Stuart Middleton
University of Southampton, UK

Stephen C. Phillips
University of Southampton, UK

Michael Boniface
University of Southampton, UK

Bassem Nasser
University of Southampton, UK

ABSTRACT

Service Oriented Infrastructures (SOIs) have recently seen increased use, mainly thanks to technologies for data centre virtualization and the emergence and increasing commercial offering of Cloud solutions. Web Services have been seen as a tool to implement SOI solutions thanks to their versatility and interoperability, but at the same time, Web Services have been considered not suitable for providing interactive real-time solutions. In this chapter the state of the art of the Web service technology will be analysed, and their different communication mechanisms and the existing implementations will be compared. Firstly, the different standardisation bodies working on Web service specifications relevant to SOI will be introduced. The various approaches to implement Web services will be described followed by the Web service specifications and the middleware that make use of those specifications, including the description of the commercial interfaces and development tools to create services for the cloud. In the last part of the chapter, the interoperability problems present on the different frameworks and the existing solutions to minimize those interoperability problems will be explained.

DOI: 10.4018/978-1-60960-827-9.ch010
INTRODUCTION

The most well known and widely used connection technology for building SOIs are web services. A web service is a software system designed to support interoperable machine-to-machine interaction over a network. There are currently two types of web services: the SOAP web services and the RESTful web services.

In the SOAP web services the interface is described in a machine-processable format, Web Service Description Language (WSDL), which provides a model and an XML format for describing web services. WSDL enables one to separate the description of the abstract functionality offered by a service from concrete details of a service description such as “how” and “where” that functionality is offered (Chinnici et al. 2003). WSDL describes a web service in two fundamental levels: the abstract level and the concrete level. Within each of these two levels, the description uses a number of constructs to promote reusability of the description and separate independent design concerns.

At the abstract level, WSDL describes a web service in terms of the messages it sends and receives; messages are described independent of a specific wire format using a type system, typically XML Schema. An operation associates a message exchange pattern with one or more messages. A message exchange pattern identifies the sequence and cardinality of messages sent and/or received as well as to whom they are logically sent to and/or received from. An interface groups together operations without any commitment to transport or wire format.

At the concrete level, a binding specifies transport and wire format details for one or more interfaces. An endpoint associates a network address with a binding. And finally, a service groups together endpoints that implement a common interface (Chinnici et al., 2003).

Other systems interact with a web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other web-related standards (Booth et al., 2004).

Web services provide a standard means of interoperating between different software applications, running on a variety of platforms and/or frameworks. Their complexity and orientation to large software vendors rather than open source software are often regarded as the major drawbacks of web services. Other criticism on web services is their performance, due to the fact of using XML as a message format and SOAP for the envelop.

The following list outlines the steps involved in a web service invocation using the SOAP approach:

1. The client application uses a client stub, which offers remote operations as seemingly local method calls, to turn its request into a proper SOAP request. This is often called the marshalling or serializing process.
2. The SOAP request is sent over a network using the HTTP protocol. The server receives the SOAP requests and passes it to a server stub. The server stub unmarshals/deserializes the SOAP request.
3. The server stub invokes the service implementation of the requested operation, which carries out the work.
4. The result of the requested operation is handed to the server stub and converted into a SOAP response message.
5. The SOAP response is sent over a network using the HTTP protocol. The client stub receives the SOAP response and unmarshals/deserializes it.
6. Finally the client receives the result of the web service operation.

The Representational State Transfer (REST) is another style of creating web service that has emerged as an alternative to SOAP implementation for web services. RESTful web services are implemented using HTTP and focused on interac-