ADJUSTMENT NEEDS IN SOA BASED E-LEARNING APPLICATIONS

Antonio Ortiz, Juan Manuel de Blas and José María Gutiérrez
Computer Science Department. Technical School of Computer Science Engineering. University of Alcala
28871 Alcalá de Henares, Madrid, Spain

Keywords: Service-Oriented Architecture, Web Services, E-learning systems, Software reuse, Interoperability.

Abstract: SOA is a promising emerging technology and as such, it still has to solve certain issues. This paper describes the solutions made to solve some interoperability issues found while carrying out an interconnection project of two SOA based systems. It is supposed that the solutions can help to avoid the problems in the future or solve incompatibilities. Not only in these projects, but also in similar ones, as the proposed solutions are not tied to a specific implementation. Both systems were developed using the same programming language, although they were based on different software platforms. These solutions are based on the analysis and the tests accomplished, where simple data structures were sent and received, using the Java language. This document also demonstrates the current state of interoperability in today's frameworks, which are not fully compliant yet, and are still rather weak using complex data structures.

1 INTRODUCTION

The Computer Science Department of the University of Alcalá, is developing an e-learning application, composed by many subsystems. Two of these subsystems, self independent, are the subject of this study.

Both systems needed a reliable communication mechanism, and it was decided to use Web Services to implement them, since there was the possibility that they used different programming languages.

Web Services architecture allows developers and corporations to encapsulate business logic, publish it as services, subscribe to other services, and share data between systems. One of the main benefits derived from using this technology is a considerable reduction in difficulty to intercommunicate along different components, offering a view of those components as a service-based architecture, fully Internet compatible.

Due to the fact that the project uses Web Services, a complete interoperability was expected. However, in the process of integration of the two subsystems, some incompatibility issues were found.

After detecting these interoperability problems, that were previously spotted by other authors (Severine, 2005), (Aragão and Fernandes, 2003) and (Söderström, 2005), it was necessary to tweak the core logic to achieve a satisfactory integration between these two systems. On the following chapters, the steps taken to gain interoperability are detailed, also describing the necessary standards and guides followed.

2 INTEROPERABILITY STANDARDS AND GUIDES

Web Services are based on the SOAP standard, with current release 1.2 (SOAP, 2003). It is an XML-based protocol used to access services on the Web. It was originally developed by IBM and Microsoft. SOAP main purpose is, in fact, similar to others object distributed systems, such as DCOM and CORBA, but less resource-consuming and easier to develop and maintain. It uses XML syntax to send messages over the Internet using the HTTP protocol. In addition, it can be concluded that because SOAP uses a simple interchange message system, SOAP can be used a messaging system as well.

Therefore, SOAP is the element that allows communication between heterogeneous systems, thanks to the use of a common language like XML. However, in order to ensure interoperability across

Ortiz A., Manuel de Blas J. and María Gutiérrez J. (2007).
ADJUSTMENT NEEDS IN SOA BASED E-LEARNING APPLICATIONS.
In *Proceedings of the Third International Conference on Web Information Systems and Technologies - Internet Technology*, pages 123-127
DOI: 10.5220/0001261901230127

platforms, operating systems, and programming languages, an organization was created. This organization is the WS-I, (Web Services Interoperability) (WS-I, 2006), which gives guidelines, creates and promotes generic communication protocols, achieving to be a referent in Web Services interoperability among different platforms and programming languages.

WS-I Basic profile is one of the most important guidelines. The version 1.0 was released on August 2003, and solved more than 200 interoperability issues. Among others, some of the standards that can be found in WS-I BP are SOAP 1.1, WSDL 1.1, UDDI 2.0, XML 1.0 y XML Schema.

Version 1.1 was released in April, 2006, and it included some improvements such as file attachment support (SwA). Some of the new requirements are described in the John Evdemon (Evdemon, 2004) personal blog, who is a specialist in business projects and workflow technologies.

WS-I also works on rules that ensure interoperability using secure SOAP messaging systems. The workgroup in charge is named Basic Security Profile Working Group, which also is in charge of some specifications developed by OASIS (OASIS, 2006). OASIS is a consortium that aids development, convergence and adoption of new standards. The more relevant of them are the ones that care about secure interoperable Web Services. It has a technical committee named Web Services Security TC (WSS), which objective is to carry on the work described on the WS-Security specification that was released in the context of Web Services Security Roadmap, published on April 2002. One of the most important specifications described is WS-Security (WS-Security 1.0 y WS-Security 1.1). This specification cares about security, using previously defined specifications and standards, avoiding defining a complete security solution, and proposing an existing set of SOAP (SOAP 1.1 and SOAP 1.2) extensions, allowing the use of secure Web Services through a great variety of security models such as Kerberos, PKI, and SSL.

It is clear that there are organizations that care about ensuring interoperability, developing rules and guidelines. Now, it is necessary to test if the uses of these guidelines are enough to make interoperable systems or something more is needed.

3 COMPLETED PROJECTS

The first system (Figure 1) belongs to the PROFIT research project "FIT-350101-2004-7", named "Educational virtual resources management and exploitation platform", which aims to create a learning management platform that will use a digital repository to store the data.

During development process guidelines from IMS DRI (DRI, 2003) were followed and the Codehaus XFire (Codehaus, 2006) platform was used for the core functions. The choice of this platform was due to its easy of use, its standard compliance, and it open source characteristic.

Figure 1: Digital Repository Interface.

The second system (Figure 2), named SROA (Learning Object Reusability System), is part of another research project: PROFIT "FIT-350101-2005-4", named "Publication and universal location of learning objects system". The goals of this project are to analyze and design a prototype system to assemble courses by reuse existing content located in diverse remote repositories.

Figure 2: SROA system interface.

The first prototypes created (Otón et. all, 2006A) and (Otón et. all, 2006B), were able to demonstrate that systems following these guides and architecture (Otón et. all, 2005), are able to distribute educative resources along many different e-learning platforms, by making their respective repositories interoperable.

IMS Abstract Framework (AF, 2003) specifications were used during the development process, and Web Services were used as well. This second system uses Systinet Server platform (Systinet, 2006), because it is easily integrated into the Eclipse development software. Systinet Server for Java offers high performance, interoperability, reliable delivery and security. This platform is widely deployed by industry leaders including Barclays Global Capital, T-Mobile and FileNet (Systinet, 2006).

Once both systems development reached an advanced stated, the integration process began, in order to build completely standalone system, as shown in Figure 3.

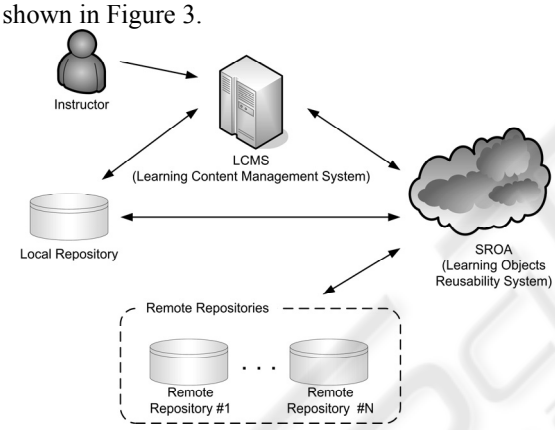

Figure 3: Scheme of the integrated system.

3.1 Observed Problems and Proposed Solutions

By using Web Services and its possibilities to communicate heterogeneous systems it is avoided the need of use of special connection mechanism for connecting different language based systems. However, the Web Services servers used were not the same and although no importance was given to this aspect, at the beginning. Later it was discovered that the integration was not seamlessly, as the results were not the expected ones. Foregoing reason, it was detected the necessity to make a higher effort in order to obtain a complete interoperability between both systems.

The first step was to ensure that the servers used fulfilled the recommendations dictated by the WS-I; information that is shown in the table 1.

Table 1: Main characteristics of the servers used.

	Systinet 6.0	*XFire 1.2*
SOAP 1.1	✓	✓
SOAP 1.2	✓	✓
SwA	✓	
WSDL 1.1	✓	✓
WSDL 2.0		
UDDI V2	✓	✓
UDDI V3	✓	
WS-I BP 1.0	✓	✓
WS-I BP 1.1	✓	✓
MTOM/XOP		✓
WS-Security	✓	✓
WS-Addressing	✓	✓
WS- R. Messaging	✓	

The following conclusions are extracted from the previous table:

- Both servers fulfil the interoperability recommendations developed by the WS-I (highlighted rows of the table). The fulfilment of these procedures should be an essential requirement for selecting a server as a platform to use for developing SOA architecture.
- They all fulfil the main standards that shape the Web Services (SOAP 1.1, SOAP 1.2 and WSDL 1.1). Only one exception is found, the XFire server, failing the SwA requirement (SOAP with Attachment), using MTOM (Message Transmission Optimization Mechanism, MTOM 2005) instead to send files. None of the servers fulfils WSDL 2.0; mainly because it is not a standard yet, only a candidate recommendation as of March 2006 (WSDL, 2006).
- About secure interoperability connections (WS-Security, WS-Addressing and WS-Reliable Messaging), it is also fulfilled by both servers, with the exception of XFire server, that does not fulfil WS-Reliable Messaging recommendation.

This information should guarantee, at least from a theoretical point of view, a certain level of interoperability, which is required when developing interoperable service oriented systems. Nevertheless, when testing the interconnection, the obtained results were not the awaited ones.

The SROA system has to perform a learning object search on the repository, and then return to

the client those objects that meet the parameters marked by the user. To send the files across the wire, a Systinet's proprietary class was used: the *ResponseMessageAttachment* class, which turned out to be incompatible with the other platform. So, the first interoperability problem was found, but it was solved using byte arrays (*byte []*). Because it is a primitive type (available in all the platforms), its use ensures interoperability.

This solution is not perfect; because another problem appears, the whole content of the file is loaded into memory. The solution to this can be to transfer it across streaming. Tests were also performed using *DataHandler* and *DataSource* classes, recommended by Sun when SAAJ is not used (SAAJ, 2006). At the end, also these classes turned out to be incompatible as well.

When returning the learning object list, the SROA system use complex objects. These objects (*JavaBean*) incorporate different attributes, such as an indicator of the operation state, or a collection (*ArrayList*) of new *JavaBean* generated in the operation. The above mentioned objects represent each of the learning objects that match the parameters of the search. These types of structures are widely used in the Java language, but they produce some interoperability problems. After trying, without success, to return an object-only collection, parameterization of the data inside was necessary, and to encapsulate it inside another object in order that at least the transmitted information was recognized across the SOAP applications that are included with the platforms.

Another problem was found sending files using SOAP. It was necessary to deactivate the MTOM (MTOM, 2005) support in the XFire platform in order to achieve a complete interoperability, because the platform used in the other project did not support it.

On the following paragraph, an example of the SOAP message sent by the platform SROA to the Repository is shown, being outlined in boldface the most relevant information. It can be observed how this information is grouped in a complex object called *DatosBusquedaBean*, including a file type. To process the sending of this file, the system decomposes in bytes (*base64Binary*), because the inclusion of it in the SOAP message once MTOM is deactivated.

```
<e:Envelope
xmlns:d="http://www.w3.org/2001/XMLSchema"
xmlns:e="http://schemas.xmlsoap.org/soap/env
elope/"
xmlns:i="http://www.w3.org/2001/XMLSchema-
instance"
xmlns:wn0="http://bean.common.busqueda.sroa.
org"
xmlns:wn1="http://xfire.codehaus.org/Servici
oWebMetadatosRepositorio">
<e:Body>
   <wn1:buscarLOEnRepositorio>
     <wn1:db
        i:type="wn0:DatosBusquedaBean">
     <wn0:ficheroXEL
   i:type="d:base64Binary">PD9lkjdfldkjgsldk
fgjsvGJDDOgjjsdfgldgorkgsdFSDJofg4wgdmuuhjsh
    … … …
JhdohjakldjvsakjhsaklvnsnaasjSSJDjsjdSJDjsjo
wekjkdsfgjlskjdfasDJDSJDSdjldvalkbnRlbnQ+
     </wn0:ficheroXEL>
     <wn0:forzarGeneracionXEL
        i:type="d:boolean">0
     </wn0:forzarGeneracionXEL>
     <wn0:opcionBusqueda
        i:type="d:int">1
     </wn0:opcionBusqueda>
     <wn0:porcentajeCoincidencia
        i:type="d:float">20.0
     </wn0:porcentajeCoincidencia>
     </wn1:db>
   </wn1:buscarLOEnRepositorio>
   </e:Body>
</e:Envelope>
```

The returned error found when the SROA system tries to send a file to a client that uses MTOM is shown in the following code. The transmitter sends the file as a stream of bytes inside the SOAP message, whereas in destiny the receiver expects this file to be out of the SOAP message.

```
Unsupported media type: application/xop+xml;
charset=UTF-8; type="text/xml"
at
com.systinet.saaj.soap.SOAPPartImpl.checkCon
tentType(SOAPPartImpl.java:486)
```

4 CONCLUSIONS

Web Services and SOA architectures, with all their presented features, were met with a great enthusiasm, because, it was possible to interconnect heterogeneous systems, as the market demanded. The possibility of using the language and the platform most adapted to every purpose, to finally obtain a complete system composed by independent elements, was finally possible. Nevertheless, and after our own experience, it is possible to conclude indicating that the obtained results are not as ideal as the announced ones by the organizations and the manufacturers.

The system integration was possible, although it has been necessary to do modifications in the transmission protocols used, and in the information sent, reducing notably its complexity. Therefore,

Web Services interoperability at this moment does not reach the degree of awaited fulfilment.

During the testing stage, the obtained results reveal that working with simple data (primitive types), guarantees interoperability. Greater problems arise when the information is more complex, for instance when using *JavaBeans*, because this information is handled in different ways depending on the platforms used.

Another detected disadvantage is the inability to use common structures in the Java environments, as the *Collection* objects. Depending on the server, it is possible to use them as a return type for the methods, being handled all in a different way by almost all of the servers, when its use is possible. It has been observed that every platform adapts the WSDL file to its own needs, resulting in incompatibilities among platforms, and producing errors while sending data, because the format of the message sent is different from to the expected by the receiver.

On top of that, with all of these issues exposed, it is expected that future works should be carried out with that limitations on mind, focusing the problems from another point of view and making easier the process of dealing the limitations.

In the future, it will be necessary for the organizations as the mentioned WS-I, OASIS, and the implementation manufacturers, to continue improving the recommendations. For example, the automation in the complex information sending (as *JavaBeans*), or the possibility of using compatible Streaming systems.

Without fulfilling with these expectations, Web Services use will be limited as occurs with other solutions on the market, such as RMI or CORBA. If this goal is achieved, the use of Web Services will be extended to all kinds of applications that need data transmission over the Internet.

REFERENCES

Severine, B., 2005. "Interoperability problems: management of evolution of collaborative enterprises". In INTEROP-ESA'05, *1st International Conference on Interoperability of Enterprise Software and Applications.*

Aragão, R., Fernandes, A., 2003. "Conflict Resolution in Web Services Federations". In ICWS'03, *1st International Conference on Web Services.*

Söderström, E., 2005. "Web services and inter-operability". In INTEROP-ESA'05, *1st International Conference on Interoperability of Enterprise Software and Applications.*

IMS DRI. 2003. IMS Digital Repositories Interoperability, *http://www.imsglobal.org/digitalrepositories/index.html*

Codehaus Xfire. 2006. *http://xfire.codehaus.org/*

Otón, S., Ortiz, A., Barchino, R. 2006A. "Arquitectura orientada a servicios Web para la implementación de repositorios distribuidos de objetos de aprendizaje". In IADIS WWW/Internet'06. *International Association for Development of the Information Society*

Otón, S., Ortiz, A., Hilera, J.R. 2006B. "Sistema de reutilización de objetos de aprendizaje". In *VIII congreso Iberoamericano de Informática Educativa.*

Otón, S., Hilera, J.R., Gutiérrez, I., Ortiz, A. 2005. "Arquitectura orientada a servicios Web para la implementación de repositorios distribuidos de objetos de aprendizaje". In SINTICE'05. *Simposio Nacional de Tecnologías de la Información y las Comunicaciones en la Educación.*

IMS AF. 2003. IMS Abstract Framework, *http://www.imsproject.org/af/index.html*

Systinet Corporation, *http://www.systinet.com/*

WSDL 2.0. 2006. Web Services Description Language 2.0 Candidate Recommendation 27 March 2006, *http://www.w3.org/TR/wsdl20/*

SAAJ. 2006. SOAP with Attachments API for Java, *http://java.sun.com/webservices/saaj/index.jsp*

MTOM. 2005. SOAP Message Transmission Optimization Mechanism, *http://www.w3.org/TR/soap12-mtom/*

WS-I. 2006. Web Services Interoperability Organization, *http://www.ws-i.org*

SOAP 1.2. 2003. Simple Object Access Protocol 1.2 Recommendation 24 June 2003, *http://www.w3.org/TR/soap/*

Evdemon, J. 2004. "What's changed in WS-I Basic Profile 1.1?".*http://blogs.msdn.com/jevdemon/archive/2004/08/25/220353.aspx*

OASIS. 2006. Organization for the Advancement of Structured Information Standards. *http://www.oasis-open.org/home/index.php*