Information Systems & Grid Technologies

Fourth International Conference ISGT'2010

Sofia, Bulgaria, May 28 – 29., 2010.

Proceedings

ISGT'2010 Conference Committees

Co-chairs

- Prof Ivan SOSKOV
- Acad Kiril BOYANOV

Program Committee

- Pavel AZALOV, Pennsylvania State University
- Marat BIKTIMIROV, Joint Supercomputer Center, Russian Academy of Sciences
- Marko BONAČ, Academic and Research Network of Slovenia
- Marco DE MARCO, Catholic University of Milan
- Milena DOBREVA, University of Strathclyde, Glasgow
- Viacheslav ILIN, Moscow State University
- Vladimir GETOV, University of Westminster
- Jan GRUNTORÁD, Czech Research and Academic Network
- Pavol HORVATH, Slovak University of Technology
- Arto KARILA, Helsinki University of Technology
- Dieter KRANZMUELLER, University of Vienna
- Vasilis MAGLARIS, National Technical University of Athens
- Ivan PLANDER, Slovak Academy of Science
- Dov TE'ENI, Tel-Aviv University
- Stanislaw WRYCZA, University of Gdansk
- Fani ZLATAROVA, Elizabethtown College

Organizing Committee

- Vladimir DIMITROV
- Maria NISHEVA
- Kalinka KALOYANOVA
- Vasil GEORGIEV

Vladimir Dimitrov Vasil Georgiev (Eds.)

Information Systems & Grid Technologies

Fourth International Conference ISGT'2010

Sofia, Bulgaria, May 28 – 29., 2010.

Proceedings

St. Kliment Ohridski Uiversity Press

Preface

This conference was being held for the fourth time in the end of May, 2010 in Sofia, Bulgaria. It is supported by the Scientific Research Fund of the University of Sofia "St. Kliment Ohridski" and by the Bulgarian Chapter of the Association for Information Systems (BulAIS). The Organizing committee consists of scientist from the Faculty of Mathematics and Informatics of the University of Sofia and from the Institute for Parallel Processing of the Bulgarian Academy of Sciences.

Total number of papers submitted for participation in ISGT'2010 was 39. They undergo the due selection by at least two of the members of the Program Committee. This book comprises 28 papers of 20 Bulgarian authors and 17 foreign authors included in one of the three conference tracks. A special section contains the selected publications of 5 young scientists. Responsibility for the accuracy of all statements in each peer-reviewed paper rests solely with the author(s). Permission is granted to photocopy or refer to any part of this book for personal or academic use providing credit is given to the conference and to the authors.

The editors

ISBN 978-954-07-3168-1
St. Kliment Ohridski Uiversity Press

Table of Contents

Intelligent Systems

Young Scientists Section

Language Compatibility Issues in Multi-Party Web Services

Sherimon Cherian [1], Dr.Reshmy Krishnan [2], Dr.Zlatinka Covacheva [3]
Higher College of Technology, Muscat, Sultanate of Oman
[1]pcsheri@yahoo.com, [2]reshmy_krishnan@yahoo.co.in, [3] zkovacheva@hotmail.com

Abstract. Web services are web-based applications developed using different programming tools that dynamically interact with other web applications using open standards that include XML, UDDI and SOAP. The web applications that involve calling multiple intermediary web services as part of an overall transaction are referred to as multi-party web services. There will be accessibility issues when accessing one web service with another one which is developed in another language because of the interoperability of the language. Because of the variety of web programming tools it cannot be avoided in accessing one web service developed in one language and another one developed in another language. For example there are issues in accessing a web service developed in Java language with another web service developed in PHP or C#. In this paper these issues have been discussed and some solutions have been suggested.

Keywords: Web Services, Java, C#, PHP, Multi-Party Web Services, Interoperability.

1 INTRODUCTION

Web services are self-contained, self-describing modular applications that can be published, located, and invoked across the Web. They represent an important evolutionary step in building Internet-based distributed applications. Web services allow buyers and sellers all over the world to discover each other, connect dynamically, and execute transactions in real time with minimal human interaction. At a minimum, web services can be any piece of software that makes itself available over the Internet using standardized web services messaging system and interface. The Web service technology comprises three basic standards:

- Web Services Description Language (WSDL): a format for describing the functionality of a service [5].
- Universal Description Discovery & Integration (UDDI): a registry that supports service publication and discovery [9].
- SOAP (formerly Simple Object Access Protocol): a protocol for exchanging messages among services.

XML, eXtensible Markup Language [4], lies at the core of web services interoperability, and enables it to provide a language-neutral and platform-independent way of linking applications [2]. Web services provide interoperability across platforms and languages. Organizations are developing web services in their

interests. It can be in J2EE or PHP or in other tools. But the client who is accessing the web service may be .NET client or Pearl client. For instance, making a call to a service checking the bank balance may in turn involve calls to multiple services before returning a response to the end user. Here each intermediary web services may be developed in different tools.

1.1 Web-services protocols

Web services are self-contained, modular applications that can be described, published, located, and invoked over the Internet. Web services architecture then requires three fundamental operations: publish, find, and bind. These operations along with the actors involved are shown in Figure 1 [3].

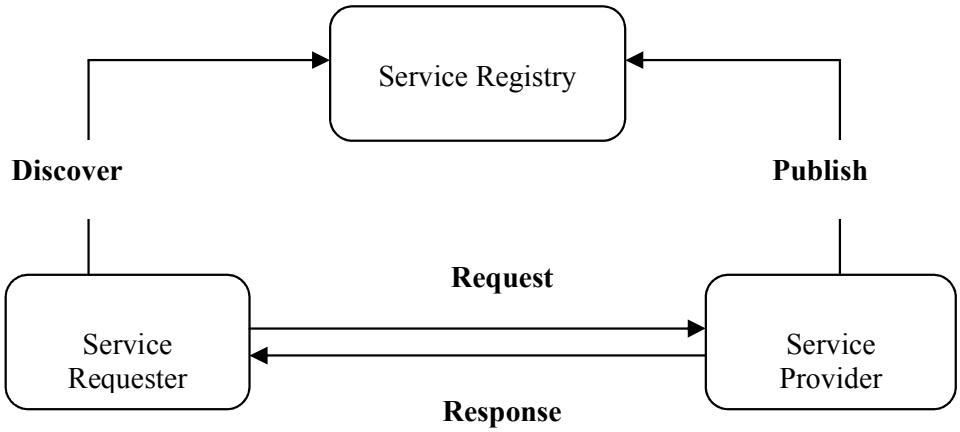

Figure 1: Web Services Actors and Operations

As shown in the figure, a service provider creates a web service and its service definition and then publishes it with a service registry based on UDDI specification. Once a service is published, a service requester may find the service via the UDDI interface. The UDDI registry provides the service requester with a service description based on WSDL. The service requester may then use this information to directly bind to the service and invoke it. All communication in the above scenario is based upon the standard messaging protocol, SOAP.

1.2 Service Oriented Architecture

Service Oriented Architecture (SOA) is a new architecture for the development of loosely coupled distributed applications. It is a collection of many services in the network. These services communicate with each other and the communication

involves exchange of data and even service coordination. Earlier SOA was based on DCOM or Object Request Brokers (ORBs). But nowadays SOA is based on web services [6].

SOA is much different from point-to-point architectures. SOA comprise loosely coupled, highly interoperable application services. These services can be developed in different development technologies (such as Java, .NET, C++, PERL, PHP etc), the software components become very reusable i.e. the same C# (C Sharp) service may be used by a Java application and / or any other programming language. WSDL defines a standard, which encapsulates the language specific implementation from the calling service.

1.3 Multiparty Web Service

The web applications that involve calling multiple intermediary web services as part of an overall transaction are referred to as **multi-party web services**. For instance, making a call to a service providing insurance quotes may in turn involve calls to multiple insurance provider services before returning a response to the end user.

1.4 Interoperable Web Services

An interoperable web service is one which can work across platforms, languages, applications and with web services from different vendors. Nowadays interoperable web services are inevitable part of web programming. As web services are shared software pieces the sharing of web services in different tools and platform are very important.

1.5 The need of Language Compatibility in Multiparty Web Services

Web Services are common and the user has the freedom to choose any web programming tool, platform, vendors to develop the web services. The interoperability can be done in several ways [7].

- Language Based: Web services developed in any tool like .NET, J2EE, PHP etc. should be able to integrate each other.
- Operating System Based: Web services developed in different operating systems like Solaris, Windows and Linux can co-exist and interoperate across assorted web environment.

Seamless integration will also enable greater collaboration for enterprises, by allowing them to leverage a larger ecosystem of partners in application development [8].

The paper is structured as follows: In Section 2, benefits & challenges to achieve interoperability are explained followed by suggestions to reduce language compatibility issues in the next section. Finally we discuss related work and conclude in Section 4 and Section 5, respectively.

2. PROFILE

In Service Oriented Architecture (SOA), and Web Services, Language Compatibility is an important factor in its success. It is generally accomplished by building the Web Services using the well-established guidelines for implementing Web Services and by following industry standards such as XML, WSDL, SOAP, and UDDI. But this is not sufficient to achieve the interoperability. The different products used for development also have to comply with many requirements such as the need to have similar implementations (data types, formats, and schemes) of the standards that you want to use. As different products are provided by different vendors, developed by several sets of people, and employ various types of underlying technologies, achieving a common understanding often becomes very difficult, which makes the products likely to be non-interoperable with each other. Over the last few years, the basic Web Services standards like XML, WSDL, and SOAP have matured a lot and WS-I have released a Basic Profile [12] that contains implementation guidelines for basic Web Services standards. Today, most vendors provide products that comply with the Basic Profile and support the standards included in the profile. With the wide adoption of the Basic Profile, software vendors have been able to make their products interoperable to a great extent.

There are four deliverables produced by the WS-I for the Basic Profile version 1.0. They are:

- Requirements and guidelines for writing interoperable Web services.
- Usage scenarios which describe fundamental ways in which providers and consumers interact.
- A sample application, which is an implementation of an interoperable Web service that demonstrates the requirements and guidelines presented in the Basic Profile.
- Testing tools which help developers to verify that their Web service implementations conform to the requirements in the Basic Profile.

Achieving interoperability for scenarios involving only basic standards is relatively easy if you follow the guidelines set by the Basic Profile (BP) 1.0 or 1.1 of the Web Services Interoperability Organization (WS-I) [12]. The Basic Profile consists of

implementing guidelines recommending how a set of core Web Services specifications should be used together to develop interoperable Web Services. The guidelines address technologies that cover four core areas: Messaging, Description, Discovery, and Security.

2.2 Benefits of Interoperability

The benefits can be summarized as follows:

- Facilitate the creation of a new presentation layer using a new technology and, at the same time, reuse the existing business components
- Integrate heterogeneous software components within an enterprise
- Lower Integration efforts
- Lower Ownership and Maintenance efforts. The cost and efforts of ownership and maintenance inherently decrease with the use of interoperable components, supplied by third parties
- Increased market and technology opportunities. With the scope of web services and interoperability, the enterprises have a wider choice of vendors, as technologies no longer remain a hindering factor

2.3 Challenges to Interoperability

As is the case with any technologies, the web service and the interoperability aspects also present the providers and consumers with a few bottlenecks. The primary reason for most of these challenges is that web service is an emerging technology. Some of the challenges faced include the following:

- One of the most intriguing challenges in interoperability is the mismatch between the data types of "interoperating" platforms. Other languages may not support the data types supported by one language. Though there may be equivalent data types, it is difficult to find interoperable data types. For example, the Arraylist type in Java is equivalent to the ArrayList type in Microsoft.NET though one cannot say they are directly interoperable. This is due to the fact the implementation of these types are specific to their parent platforms [11]
- Every organization tries to address the interoperability in their own perspective, giving rise to "proprietary" standards, which defeat the very objective of web services and interoperability. These specifications cannot be guaranteed to be compatible with those adopted by other vendor(s) or client(s)
- Lack of any industry standards and/or specifications

- With the growing scope and horizon of the web services, new specifications and standards are being constantly added to the existing set by the vendors, consortia of companies, standard bodies as well as individual companies. These groups may or may not be communicating with each other regarding any of their releases. This contributes to a growing set of incompatible specifications

The best way to counter the diverse specifications and surmounting barriers to the web services interoperability is through consensus and industry-accepted and guided specifications. This shall be the only way that the enterprise end-users are assured of interoperable web service solutions.

3. SUGGESTIONS FOR DEVELOPING WEB SERVICES TO REDUCE LANGUAGE COMPATIBILITY ISSUES

The following are some suggestions to solve the language compatibility issues between web services developed in Java and in Microsoft .NET.

Check for Empty Arrays

If an array of objects is sent over a web service, always ensure that the array contains valid data [10]. Sending empty arrays over web services can create issues because some tools consider an empty array as a single null value.

Generation of Client Proxies

Many Java based tools have the option of specifying a unique package and type name when generating client proxies. This is essential when you are creating proxies for Web Services that share the same data type.

Java Beans

Ensure that an object is not null when an IDE is used to generate Java Beans from an XSD file.

Null Dates and Times

In Java, Java.util.Date and Java.util.Calendar are classified as reference types. In .NET, System.DateTime is considered a value type. Reference types can be null, whereas value types cannot. If you are planning to send null date values across a Web Service, always send the value in a complex type, and set the value of the complex type to null. This will help prevent the null date value being interpreted incorrectly (and raising an exception).

compareTo() method

If sending dates and times over a Web Service between .NET and Java, always use the appropriate compareTo() method in Java to compare dates (as opposed to date == value). This will help ensure accuracy for date comparisons between the platforms.

Trace Tool

A Trace Tool can be invaluable for investigating SOAP requests and responses between Web Services. It can help validate data types and the construct of the message, and also report SOAP faults that you may miss in a browser. Using one that intercepts the request is more difficult to setup, but easier to use than looking through trace files.

Option to Change Host and Port

When designing your Web Service client, consider adding a helper method to change the host and port values of the Web Service location. This makes it easy if the location of the Web Service changes in the future or you want to redirect the output to a trace tool.

Ensure Document/Literal when generating Web Services

Many toolkits have the option of choosing either RPC/Encoding or Document/Literal as the default format when generating Web Services. To help ensure compliance with the WS-I Basic Profile 1.0, select Document/Literal as the default encoding mechanism for all of your Web Services.

Use Unit Tests to Test Interoperability

Unit tests (using NUnit for .NET and JUnit for Java) are invaluable for testing interoperability of multiple data types over a Web Service. Re-running the tests if data types change (or if you change versions of Web Services toolkits!) will give you the confidence that the Web Services that you design are fully interoperable.

Use XSD First

When designing for interoperability, always start with defining the data first. Deciding on what data will be sent, creating the data type in XSD first, and then using tools to generate classes from the XSD file will help guarantee data type interoperability.

4 RELATED WORK

It was the challenges, as described above, that forced the technology leaders in Web Service specifications to collaborate and create the Web Services Interoperability Organization (WS-I) [12]. The organization was founded in 2002, and is aimed at promoting interoperability across platforms, applications and languages. The charter of the WS-I is to accelerate the adoption of Web Services by assisting in the selection and interpretation of WS specifications and in the development of common best practices for development and deployment and integration of business applications.

The organizations joining the WS-I can benefit from the following perspectives:

- Since the WS-I is collaborated on the basis of use-cases derived from industry specifications, it acts as a forum for communicating the business requirements and those for distributed computing interoperability.
- The organizations having proven credentials in a particular vertical, like Insurance and Hospitality, can help leverage and augment the horizontally focused computing with vertically focused business requirements.

Apart from the access to WS-I profiles, the member also gains access to the testing methodologies for web services, the usage & implementation scenarios and use-cases for the same. All these contribute to a reduced cost in development, deployment and integrations of their service-oriented solutions.

5. CONCLUSION

Web services greatly help developers build highly integrated solutions. Web services are certainly going to form a huge chunk of the enterprise deployments in near future. The deciding factor in their success and acceptability will be the extent of interoperability provided to the enterprise partners. To achieve this, a cohesive interpretation of ambiguities, incompatibilities and best practices is needed. These are precisely the lines that the WS-I is functioning on. In this paper issues involved in web services interoperability between Java and Microsoft .NET are discussed and some solutions have been suggested. So in order to keep Web services interoperable, we need to ensure that various vendor implementations are interoperable, the tools that are used do not affect the integrity of the Web services, and standards are strictly followed and implemented.

REFERENCES

[1] Chris Kaler (Editor). WS-Security, Version 1.0. An IBM, Microsoft and VeriSign joint specification. April 5, 2002.
http://www-106.ibm.com/developerworks/webservices/library/ws-secure/

[2] Don Box (DevelopMentor) SOAP: Simple Object Access Protocol 1.1 W3C Note. May 8, 2000.
http://www.w3.org/TR/2000/NOTE-SOAP-20000508/
[3] K.Gottschalk, S. Graham, H. Kreger, and J. Snell Introduction to Web services architecture. IBM Systems Journal, Volume 41, Number 2, 2002
[4] Jayavel Shanmugasundaram1, Eugene Shekita1, Rimon Barr2, Michael Carey3, Bruce Lindsay1,Hamid Pirahesh1, Berthold Reinwald1 Efficiently publishing relational data as XML documents , The VLDB Journal (2001) 10: pp.133–154
[5] Eyhab Al-Masri and Qusay H. Mahmoud Department of Computing and Information Science University of Guelph, Guelph, ON, N1G 2W1 Canada, Investigating Web Services on the World Wide Web, WWW 2008 Web Engineering - Web Service Deployment Beijing, China
[6] Service Oriented Architecture, http://www.service-architecture.com/
[7] http://www.sun.com/smi/Press/sunflash/2005- 11/sunflash.20051104.1.xml
[8] Common Language Resources and Technology Infrastructure, http://www.clarin.eu
[9] UDDI Version 3.0.2. http://www.uddi.org/pubs/uddi_v3.htm.
[10] Arun Gupta, Project Tango: Adding quality of service and .NET Interoperability to the Metro Web Services stack: http://blogs.sun.com/arungupta
[11] Narayana Rao Surapaneni, Dhananjay Khatre, "Java and .NET a developers guide to Interoperability and Migration, Prentice Hall India"
[12] Web Services Interoperability Organization. http://www.ws-i.org/