GENERAL INFORMATION AND MAIN CHARACTERISTICS REGARDING WEB SERVICES. PROTOCOL SOAP AND REST

Pîrnău Mironela
"TITU MAIORESCU" University, Faculty of Science and Technology Information, Bucharest, Romania; 22, Dâmbovnicului Street, sector 4, 040051,Bucharest, Romania mironelapirnau@yahoo.com, tel:0723210205

A web service represents a communication interface offered by the server, through that the clients (programs on other systems) may require different information. The customer may vary, may be present on the same computer server, may be in the same local network or it may be located in the opposite part of the country. It is a method by which applications can communicate with each other through asynchronous messages or calls on remote procedure (RPC Remote Procedure Call). So it can be concluded that a Web service is a software application that can be accessed remotely using XML-based languages. Web services are just two programs that change data between them on the Internet or Intranet in XML format called SOAP (Simple Object Access Protocol).

Key-words: Web Service, B2B, protocol, XML, SOAP, REST.
Cod JEL: M15

Introduction

The Web services are based exclusively on the W3C standards that define the SOAP (Simple Object Access Protocol), a XML message structure for transporting the coding scheme of the message body, WSDL (Web Services Description Language) the description language of the service, which defines the service and its connections with various schemes for transporting and coding the message, UDDI (Universal Discovery, Description and Integration) that provides a distributed deposit for 'traffic' of the definitions service. Because a web service can be described in WSDL, Axis has included a feature that produces the WSDL file for the web service.
http://localhost:8080/axis/NumeServciu?wsdl
Having the WSDL file of a web service it is very simple to write the programmed client to access the web service. In the programmed client the following must be changed:
1. URI for the web service;
2.The method name; 3.The input and output parameters;
The Web Services should be easily extended and reused in new applications; this is achieved by adopting the orientated programming object as well as for the modeling usage. The Web service must offer interoperability regardless the platform, the operating system and programming language, the problem that was solved by using XML language (extended Markup Language) used in order to transmit data through the network. Because XML is a very general vocabulary, a customization has been decided, achieving the SOAP result (Simple Object Access Protocol) in a role of imposing a set of rules for formatting the XML-message that contains the transmitted information. All Web services must contain a standard of Open Standard type in order to allow the communication between written components in different languages or existing on different platforms. The costs for realizing a Web service are low, because based they are based on an already infrastructure, formed by the communication network, the protocols used these, etc., Web services allow the communication *between B2B applications*. Running a service can be made independently of platform (standalone) or in the context of a server. Running the independent service of the platform is allowed by the .Net platform. In reality it is not entirely independent. This type of service is composed of two main classes: the class that contains the interface offered by the service (as well as the implementation) and the class that launches in execution the service. This class is in reality the server; it has a standard form and announces the .NET platform that wishes to register a service to a certain address of the certain system and port. The functioning of these classes follows the stages:
1) The .NET platform receives the request from the client in the form of a SOAP message and sends it forward to the Web services register;
2) The register de-serializes the message through SOAP engine;
3) Based on a decoded message, the register loads the service desired by the customer and appeals the required method;
4) The register receives the result of the requested method;
5) Register serializes the result of the method through the SOAP engine;
6) The register sends to the client the serialized result in a SOAP format.
Running the service through a server allows the configuration of certain files to locate (deploy) and removal (un-deploy), of the service. Such routine operations to take place automatically. These special files are specific to each server separately. For this type of running it is used the running on a server supporting services (for example ll represents IIS and the .NET services) or the running the service through a web application installed on an application server (e.g. Apache AXIS running on a Tomcat or Orion server). The solution most commonly used for servers that support services is the IIS combination with .NET services (offered by Microsoft) or Apache server with the .Net module. In these cases, the server has available (usually extern) facilities offered by the .NET

platform, meaning the SOAP engine is not capsuled into the server. Unlike the standalone variant, in this case the server is responsible for managing services and running them replacing the poor variant with a Web Services Registry. By installing an application server (for example Tomcat), inside it will be installed a Web application which takes the place of the services server. Through this application it is made the management of services installed and the loading and their execution.
The route that a SOAP request follows, in this case is: The application server receives a request from the client. It has no relevance what kind of demand it is (it could be HTTP Post from a script page or a SOAP message). Based on the addressed URL, the message is sent to the Web application that is registered at that location. We suppose that we got a SOAP message addressed to the Web Server SOAP application installed on the server.
In this case the message is sent to the entry point of the application (in our case the processor of demands that contents the Server SOAP application). The requests processor de-serializes the message using the SOAP Engine contained by the Server SOAP application. The contents processor locates the service that must be used, loads it and executes the method required by the client. The application processor receives the result of the executed method. The requests processor serializes the result of the method using the SOAP engine. The demands processor application sends the application server the answer already serializes. The applications server sends to the client the reply.

Tools used for Web services
Since not all application servers support the installation of the SOAP service it is required the usage of a secondary application which aims the implementation of the functionality required by the SOAP specifications. There exist many utilitarian programs aimed the creation and the maintenance with minimal effort of the web services such as:
- Apache SOAP Developed by Apache Organization;
- Apache AXIS. Developed by Apache Organization, the successor project of SOAP. It offers much more functionality, rebuilt design, working speed and better use of the memory;
- Net Platform. Developed by Microsoft. It could also be found a Ximian implementation and for Linux operating system;
- Visual Net. Developed by Antarctica. The module for Apache Server, offers among others and .Net services.
- kSOAP. Developed by Enhydra. It provides support only for the client.

SOAP Protocol Characteristics
The SOAP protocol is destined to develop applications based not only on simple applications of B2B or eCom. and it is a standard method of the technology of infrastructure for the calculation distributed by the multi platform based on XML. The SOAP technology is focused on common issues to all scenarios of the distributed calculation and follows the mechanisms: a mechanism for defining the communication unit that capsulates any message sent in an envelope with a pre-settled structure ;an extensible mechanism with the role of adding new functionalities, a and a flexible mechanism for data representation, allowing exchange of data already formatted (text or XML) and conventions for the representation of abstract data structures in a programming language mechanism, a convention mechanism for RPC (Remote Procedure Calls)representation involving definition of structures to a standard procedure call and remote procedure for sending the response, a linked mechanism for the SOAP messages related to HTTP, which is the most used communication protocol.
SOAP messages are encoded in XML documents being comprised of a wrap (SOAP envelope - demanded), a header (SOAP header - optional) and a body (SOAP body - required). SOAP messages are also XML documents, the protocol specifications providing details for a standard strong encryption of data in SOAP messages. Figure 1 contains the structure of a SOAP message.

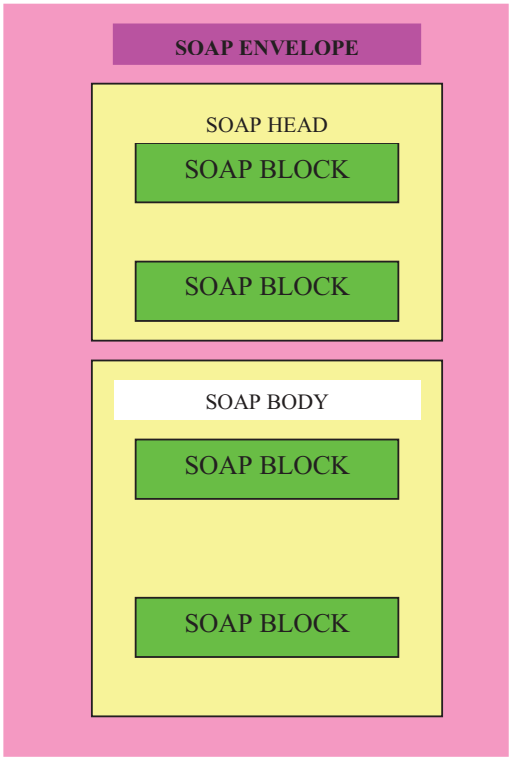

Figure 1. SOAP Message Structure

The SOAP message header is a primary mechanism extension of the SOAP protocol functionality. The specifications require only headers sent through the certain area to be XML valid elements, without adding other restrictions.
The header area is represented by the SOAP-ENV: Header and it is optional. If present, it must be the first child of the of SOAP-ENV element: Envelope and includes any number of entries (called headers). So a header is a XML element is a direct descendant of the SOAP-ENV element: Header.
The elemental SOAP-ENV:Body contains directly information representing the base of SOAP messages. All the children (descendants) of the direct element Body are called bodies. The bodies may contain arbitrary XML such rules are specified in connection with calls to the remote procedure and they will be presented in one of the following subchapters. The SOAP specifications contain information referring to the model that the objects are included in the SOAP XML language.

REST architectural model
- REST stands for the Representation State Transfer and is an architectural model for creating web services. REST describes an architecture oriented on resources. REST applies the web architecture services web, it is not a standard, it uses the standards:
- HTTP (Hypertext Transfer Protocol)
- URI (Uniform Resource Identifier)
- XML /HTML/GIF/JPEG

Basically REST supposed to build a web service using HTTP, XML and URI as it was built and the web. In a REST architecture the data on which the client tells the server to operate are in the URI and the operation server makes the data to be described directly by the HTTP method. From a technical point of view, the REST type architecture is described by the notion of “resource”, “URI”, “representation”, “uniform interface”. Binding these components in a more effective manner results in the creation of a REST architecture, presented in figure 2.

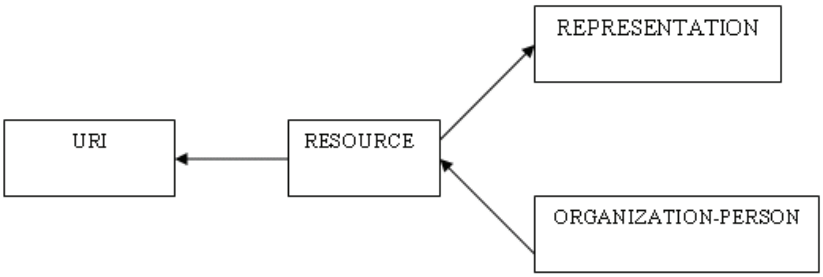

Figure 2. Items of REST architecture

In a REST architecture everything represents a resource. Everything may be referred to as an object is a resource. In general, everything that can be stored in the computer and represented as a stream of bytes is a resource. Each resource is associated with a URI. which represents the name and address of a resource. URI is the short term of the English "Uniform Resource Identifier", an identifier of a resource on the Internet, such as a document or a website. Often the URI of a resource is the same with its URL, the URL is short term for "Uniform Resource Locator", an early identifier. If some information does not have an URI then it is not a resource and is not practical on the web. An URI of a resource should be descriptive.

The "Uniform interface" is the basic principle of REST. In the entire web there are few operations which can be made on a resource. HTTP provides four basic methods for describing the most common operations:

The receipt of the representations of a resource: HTTP GET

Creating a new HTTP PUT resource for a new URI or HTTP POST to an already existing URI - The modification of an existing resources: HTTP PUT - The deleting of an existing resource: HTTP DELETE

The REST architecture is one without constraints and allows calls to functions, methods invocation, distance call procedures, such as and other messages that are understood by a certain server or a small subset of the components of the architecture.

Conclusions

The multitude of protocols and available standards from the end of the last century in the sphere of the Internet have enabled the possibility of communicating between applications on the system at large distances, with Internet access. Thus, there are systems that provide information services and information processing that in general are independent of the hardware platform, the access to them is made through the web services. A web service is a collection of protocols and standards used for exchanging data between applications or systems. Software applications written in different programming languages and running on various platforms can use Web services to exchange data network (Internet), in a manner somehow similar to inter-process communication on a single computer. The interoperability is due to the usage of adequate public standards. Based on XML, Web services have allowed the definition of standards and widely accepted technology for the IT industry. The web services have become today a necessity because it simplifies a lot the Internet by providing a distributed architecture. Basically, now, a company is not obliged to implement each of its already existing applications but it can access a web service to obtain the desired information. It results the re-usage of web services. Web services have a modular architecture, becoming a scaling one. The RPC (SOAP) architecture is a complex architecture that uses many protocols to make the connection between the client and the web service, but at the same time is more easily used by the client for it has a well-defined contract. Google uses SOAP type services. REST architecture is a simple one, but REST services, is more difficult to use by the client. The client must understand the uniform interface of the service. All services from Yahoo use REST. Any chosen architecture must be well documented for an easy usage by the client.

Bibliography

http://www.altova.com
http://aws.amazon.com/
http://en.wikipedia.org/wiki/SOAP
http://webservices.xml.com/
http://tomcat.apache.org/
http://www.codeproject.com/
http://www.w3schools.com
http://www.w3.org
www.xfront.com
http://www.xfront.com/files/tutorials.html
http://www.soapware.org/directory/4/services
http://searchsoa.techtarget.com/tips/0,289484,sid26,00.html