Business documents presentation and interchange using SOAP

Ivan Magdalenić[1], Ivo Pejaković[1], Dražen Pranić[2]

[1]Faculty of Electrical Engineering and Computing
University of Zagreb
Unska 3, Zagreb, Croatia
Tel.: +385 1 6129 756 E-mail: ivan.magdalenic@fer.hr

[2]Agrokor d.d.
Trg Dražena Petrovića 3, Zagreb, Croatia
Tel.: +385 1 4894 162 E-mail: drazen.pranic@agrokor.hr

Abstract – Simple Object Access Protocol (SOAP) is a lightweight and simple XML based protocol that is designed to exchange structured and typed information on the Web. The purpose of SOAP is to enables rich and automated Web services based on a shared and open Web infrastructure. SOAP massages are XML documents defined in a mandatory SOAP envelope. A Graphic User Interface (GUI) has been created for SOAP message transaction example. XML message is present in the GUI as directory tree as this form is familiar to the end user. Each node in XML message is represented as node in directory tree. SAOP messages are sent and received using http and https protocol.

1. INTRODUCTION

The most enterprises must cooperate with many other enterprises to be able to complete their mission. Currently, there are no worldwide-accepted standard for communication between companies. Communication is mostly point-to-point type. That means that communication between company and every company's partner is specific and unique. Data format and naming is also specific for every communication.

The best approach in this situation is to use some universal standard for describing data. Extensible Markup Language (XML) is the best choice for this task. *XML documents are made up of storage units called entities, which contain either parsed or unparsed data. Parsed data is made up of characters, some of which form character data, and some of which form markup. Markup encodes a description of the document's storage layout and logical structure. XML provides a mechanism to impose constraints on the storage layout and logical structure* [2].

SOAP provides a simple and lightweight mechanism for exchanging structured and typed information between peers in a decentralized, distributed environment using XML [1]. SOAP is sponsored by industry giants Microsoft, IBM, and others. XML and SOAP are supported as communication technology by most of business world. XML and SOAP have important role as integral part of ebXML standard.

In this paper, we will first describe XML and SOAP. SOAP will be described as part of ebXML standard. The application for presentation and exchanging SOAP messages are presented and described. Finally, communication between SOAP server and SOAP client is presented.

2. SOAP

Today's applications mostly communicate using remote procedure calls (RPC) between objects like in Common Object Request Broker Architecture (CORBA) or Distributed Component Object Model (DCOM) environment. Unfortunately, HTTP isn't designed for RPC and that is a reason why RPC aren't easily adaptable to the Internet. In heterogeneous environment like the Internet RPCs represent compatibility and security problem (firewalls and proxies will normally block this kind of traffic). On the other hand HTTP is supported by every software or hardware platform on the Internet and it would be very useful to use it for communication between applications. SOAP is designed to accomplish this. SOAP allows applications on the Internet, which is distributed and heterogeneous environment, to communicate without taking care about operating systems or hardware platforms they are running, different technologies they are built or program languages they are programmed.

According to version 1.1 of SOAP specification (currently W3C is working on version 1.2) SOAP consists of following tree parts:

- The SOAP envelope construct which defines overall framework for expressing what is in a message, who should deal with it and whether it is optional or mandatory;
- The SOAP encoding rules defines a serialization mechanisms that can be used to exchange instances of application-defined data types;
- The SOAP RPC representation defines a convention that can be used to represent remote procedure calls and responses.

Every SOAP message is encoded in XML and contains following elements:

- "Envelope" element (mandatory), which is root element of SOAP massage and defines content of message;
- "Header" element (optional), which contains header information;
- "Body" element (mandatory), which contains call and response information.

The Envelope element must be the first element in SOAP message and it defines XML document as a SOAP message.

The Header is a generic mechanism for adding features to a SOAP message in a decentralized manner without prior agreement between the communicating parties [1]. Extents of SOAP message that can be implemented in the Header are transaction processing, security, payment etc. If it is present in the message, it must be the first immediate child element of the Envelope element. The Header element can contain child elements but they must be immediate child elements of it.

The Body element contains information that is passed from sender to receiver (RPC parameters, application data etc.). The Body element is immediate child element of the Envelope element and if there is a Header element in message then a Body element comes immediately after it. The Body element can contain child elements but they must be immediate child elements of it.

Within the Body element may be the Fault element, which contains information about errors that have occurred and/or status information. If present, the Fault element must not appear more than once within the Body element.

SOAP massage must not contain Document Type Definition (DTD) or any processing instructions.

Structure of SOAP message is shown in figure 1.

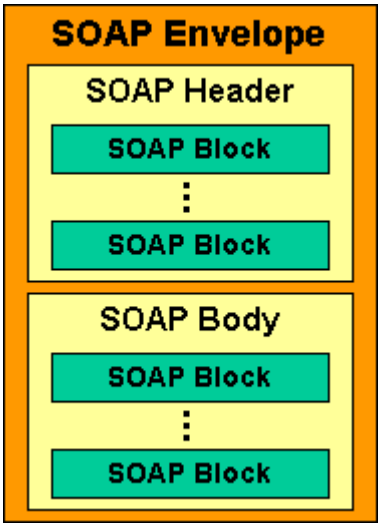

Figure 1. Structure of SOAP message

SOAP specification defines two namespaces:

- The SOAP envelope has the default namespace: http://schemas.xmlsoap.org/soap/envelope;
- The SOAP encoding and data types have the default namespace: http://schemas.xmlsoap.org/soap/encoding.

The SOAP encoding defines rules of expressing application-defined data types in XML. It is based on W3C XML Schema definition. A value may be simple or compound constructed as a composite of several parts each with a type. Simple values are built-in types from XML Schema Part 2 (simple types, enumerations, arrays of bytes). Compound values may be structs, arrays or complex types.

Binding SOAP to HTTP provides the advantage of being able to use the formalism and decentralized flexibility of SOAP with the rich feature set of HTTP. Carrying SOAP in HTTP does not mean that SOAP overrides existing semantics of HTTP but rather that the semantics of SOAP over HTTP maps naturally to HTTP semantics [1]. With "SOAP binding" we assume set of rules for carrying a SOAP massage within or on top of some other protocol for the purpose of transmission. Typical SOAP bindings are carrying the SOAP massages within HTTP or on top TCP.

SOAP follows request/response massage model of HTTP. Examples of the SOAP requests and responses are shown in figure 2. and figure 3.:

```
POST /StockQuote HTTP/1.1
Host: www.stockquoteserver.com
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAPAction: "Some-URI"

<SOAP-ENV:Envelope
  xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelop
e/"
  SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.or
g/soap/encoding/">
   <SOAP-ENV:Body>
       <m:GetLastTradePrice xmlns:m="Some-
URI">
           <symbol>DIS</symbol>
       </m:GetLastTradePrice>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figure 2. SOAP request

In the example above SOAP request is used in combination with HTTP POST method. A SOAP request can be used with any other HTTP method. Status of the SOAP response is 2XX in case of the successful receipt and 5XX in case of the error. The SOAP message with the status indicating error (5XX) must contain the Fault element within the Body element, which indicates the SOAP processing error.

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn

<SOAP-ENV:Envelope
  xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelop
e/"
  SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.or
g/soap/encoding/"/>
   <SOAP-ENV:Body>
       <m:GetLastTradePriceResponse
xmlns:m="Some-URI">
           <Price>34.5</Price>
       </m:GetLastTradePriceResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figure 3. SOAP response

One of the major design goals of SOAP is encapsulation of RPC calls into HTTP massages using flexibility and extensibility of XML. A RPC call maps naturally to a HTTP request and a RPC response to a HTTP response. According to the SOAP specification using SOAP for RPC isn't limited only to the HTTP protocol binding. To make RPC call following information is needed:

- The URI of the target object;
- A method name;
- Method signature (optional);
- Method parameters;
- Header data (optional).

All information for RPC method calls and responses are carried within the SOAP Body element. Applications may process requests with missing parameters but may also return fault. A response message must not contain both result and fault. Any additional information that method may need and it is not part of method signature may be embedded within The Header element (transaction ID etc.).

3. SOAP AND ebXML

SOAP's importance extends beyond its definition of an XML-based message protocol. Several other e-business specifications based on XML use SOAP for its messaging functions (ebXML, BizTalk, Universal Description, Discovery and Integration)[5].

EbXML is a vendor-neutral standard, developed to support SMEs (Small and Medium size Enterprise) by facilitating low cost, easy-to-install implementations that are easy to support. EbXML messaging service wholly relies on SOAP.

In figure 4. is shown how SOAP is integrated into ebXML messaging service specification.

SOAP in its original form did not support non-XML attachments. Some e-business messages, however, will carry non-XML binary files, such as images. That problem is solved with SOAP With Attachments specification. SOAP and specifically the SOAP With Attachments specification, submitted to the W3C as a Note in December 2000, defines a SOAP package that combines the SOAP 1.1 message with a MIME envelope to include direct attachments or references[6].

The ebXML header container consists of a SOAP message with a SOAP header and SOAP body. The SOAP header includes the traditional functions found in business message headers, such as identification of the parties to the transaction (sender, receiver, copies).

The SOAP body, still within the overall message header container, carries data cataloging the message contents, which is called a manifest in ebXML parlance [9].

The ebXML body container carries the payload that can be in XML or any other digitized format[7].

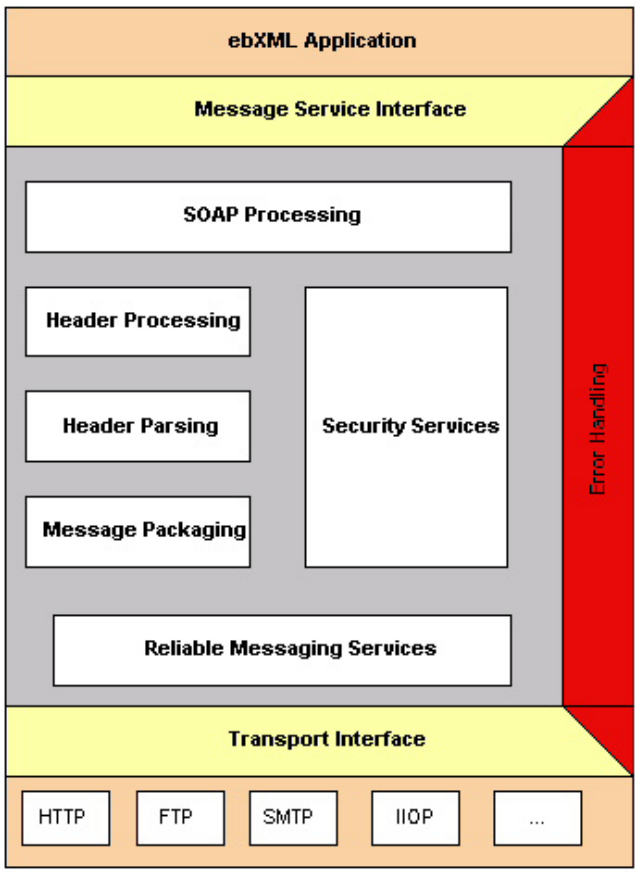

Figure 4. EbXML messaging service and SOAP

4. DOCUMENTS INTERCHANGE

Intention is to present communication between companies. There are many available implementation of SOAP server. SOAP server made by Apache XML project group was chosen because is made in JAVA [3]. JAVA is chosen as programming language, as it is platform independent.

Standard three-tier architecture was adopted. Database tier, founded on *Oracle 8i* technology, is used for storing all business data. Application tier consists of servlets deployed in application and web server Tomcat v. 3.2 [4]. Client tier can by command prompt or as in our case special design GUI [8].

Basic architecture of SOAP server used in our sample communication is shown in figure 5. SOAP server is implemented through two main servlets. First one is called RPC router and it is answerable for administration of SOAP server. Second servlet is called Message router. Basic difference between RPC router and Message router is in incoming XML messages format. XML message sent to Message router must have embedded processing instruction (purchaseOrder tag in Figure 6.).

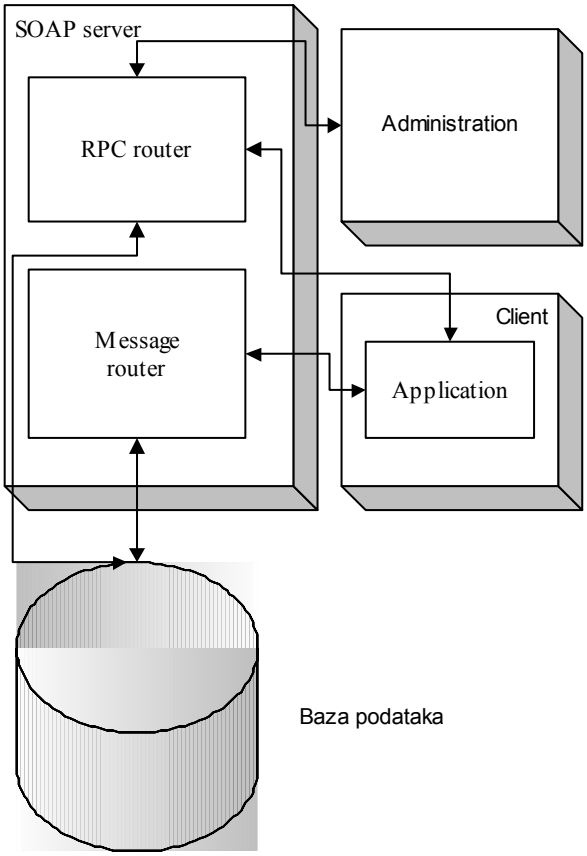

Figure 5. SOAP server architecture

Processing instructions are responsible for rising appropriate process on server. XML messages sent to RPC router demands special interface on client s tier. Naturally, input to this kind of interface can be data in different form such as XML (Figure 7.).

```
<s:Envelope
xmlns:s="http://schemas.xmlsoap.org/soap/env
elope/">
<s:Body>
<purchaseOrder xmlns="urn:po-processor">
    <items>
        <item partNum="872-AA">
            <quantity>1</quantity>
        </item>
    </items>
</purchaseOrder>
</s:Body>
</s:Envelope>
```

Figure 6. Sample XML input to Message router

```
<items>
    <item partNum="872-AA">
        <quantity>1</quantity>
    </item>
</items>
```

Figure 7. Sample XML input to RPC router

SOAP server brings us platform for XML documents interchange. However, additional software must be included for every task.

Our software is made of two parts. Server part consists of one or more JAVA classes. Those classes are receiving and handling requests. After our classes are added to classpath of SOAP server logical mapping is required, which is done through application deploy. During application deployment logical requests are specified intending to appropriate JAVA class.

Our choice is communication using “Message router” because we can focus on developing universal interface for sending SOAP messages. JAVA class is deployed on server with physical name “POProcessor” and logical name “urn:agrokor-po-processor”. Deployment is possible using web interface (Figure 8.) or using command line call.

Figure 8. SOAP Administration web interface

```
DeploymentDescriptor.xml
<isd:service
xmlns:isd="http://xml.apache.org/xml-
soap/deployment"
id="urn:agrokor-po-processor"
type="message">
<isd:provider type="java"
scope="Application"
methods="process po 007 XXX">
<isd:java
class="hr.agrokor.soap.POProcessor"
static="false"/>
</isd:provider>
<isd:faultListener>org.apache.soap.server.DO
MFaultListener</isd:faultListener>
</isd:service>
```

Figure 9. Deployment parameter using command line

Deploying application on SOAP server means enabling software to send and receive SOAP messages.

Deployment using web interface means describing connection between JAVA class that is located in SOAP server classpath and logical name of this class.

In our case class “POProcessor” is accessible outside server only by its logical name “urn:agrokor-po-processor”.

The same thing can be done deploying application using command line call. First “Deployment Descriptor” XML file must be created. “Deployment Descriptor“ contains data like physical and logical name of classes. It also provides names of methods available to call (Figure 9.). Command line call for deploying application is shown in figure 10.

```
java
org.apache.soap.server.ServiceManagerClient
http://localhost:8080/soap/servlet/rpcrouter
deploy DeploymentDescriptor.xml
```

Figure 10. Command line call for application deployment

Client part of application is made in Java. GUI is made with following features:

- Loading XML documents from file system (Figure 11.)
- XML documents representation as directory tree (Figure 12.)
- Sending SOAP message using HTTP or HTTPS
- Adjustable target URL
- Saving received SOAP message (Figure 11.)

Figure 11. Loading and saving XML documents

Standard dialog box are provided for loading and saving XML documents. Currently, GUI can only view XML file but changes are not possible. Some text or XML editors can serve to create or to modify XML documents.

The basic idea of the presentation of XML documents as directory tree is that this form is familiar to the end user. Each node in XML message is represented as node in directory tree (Figure 12.). Our intention is to build such interface that can provide same level usability as the one provided in windows system. That will include a validation of the XML document with appropriate XML scheme.

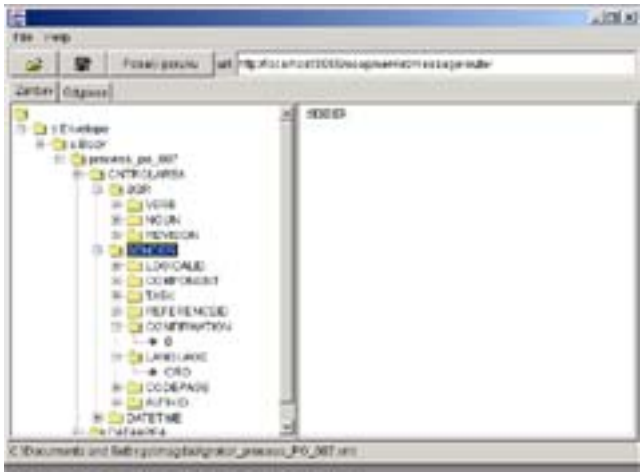

Figure 12. XML document presentation as directory tree

Using above described GUI, communication between two virtual companies has been simulated.

Best results can be given with integration of SOAP with established business process and application. Communication can be organized in such a way that minimum of human interaction is required.

5. CONCLUSION

With introduction of XML started a new period of data standardization in different business areas. Main purpose of SOAP is interchange of the XML documents, although a content of a SOAP message does not need to be an XML document. Presentation of the XML documents is in a form of directory tree as this form is familiar to the end user. This paper describes basic features of SOAP and first steps needed for adoption of this technology.

REFERENCES

[1] www.w3.org/TR/SOAP/

[2] www.w3.org/TR/2000/REC-xml-20001006

[3] xml.apache.org/soap/

[4] jakarta.apache.org/tomcat/

[5] www.xml.com

[6] http://www.w3.org/TR/2000/NOTE-SOAP-attachments-20001211

[7] ebXML Ropes in SOAP, Alan Kotok, April 04, 2001. www.xml.com

[8] Marko Topolnik, Damir Pintar, and Mihaela Sokić: Experimental Implementation of Emerging e-Business Technologies: ebXML and PKI, unpublished, 2002.

[9] Mihaela Vrhoci, Dubravka Đelmiš, Jarmila Maksimović,*Collaboration-Protocol Profile (CPP) and Collaboration-Protocol Agreement (CPA) in B2B transactions,* unpublished, 2002.