SOAP – "Simple Object Access Protocol"

Preface

It is important for application development to allow Internet communication between programs. Today's applications communicate using Remote Procedure Calls (RPC) between objects like DCOM and CORBA, but HTTP was not designed for this. RPC represents a compatibility and security problem; firewalls and proxy servers will normally block this kind of traffic. A better way to communicate between applications is over HTTP, because HTTP is supported by all Internet browsers and servers. SOAP was created to accomplish this. SOAP provides a way to communicate between applications running on different operating systems, with different technologies and programming languages.

This document has been written for Self-Study purposes which accounts for two credits in module Service Oriented Architecture (SOA), WAT 5101, at the University of Technology Mauritius. Its focus is on the SOAP 1.1 specification, which was submitted as a W3C Note by UserLand, Ariba, Commerce One, Compaq, Developmentor, HP, IBM, IONA, Lotus, Microsoft for describing services for the W3C XML Activity on XML Protocols on the 8th of May 2001. (Note that SOAP 1.2 is a W3C recommendation as from 27 April 2007).

Introduction

We'll start with an analogy between HTTP (HyperText Transfer Protocol) and HTML (HyperText Markup Language). HTML is used to build web pages using a set of tags that provide formatting for plain text. If you wanted someone to view your webpage, you could simply save and send it in a disk or as an attachment via email. However, this is not how the internet works (people exchanging disks with HTML pages). There is a method that sends you an HTML page when you type in an address in your web browser: the HyperText Transfer Protocol. HTTP provides a communication mechanism (protocol) for one computer to send HTML to another computer across the Internet.

In the same way, XML provides a way to give meaning to a set of data, using markup tags. This data can be used by different applications found at different locations. One application can generate data or XML as much as it wants, but without a person feeding the results into another application, the two would not be able to communicate with each other. There has to be a mechanism (protocol) that allows those applications to communicate, this is where SOAP steps in. Hence, one application can view another across the internet with SOAP.

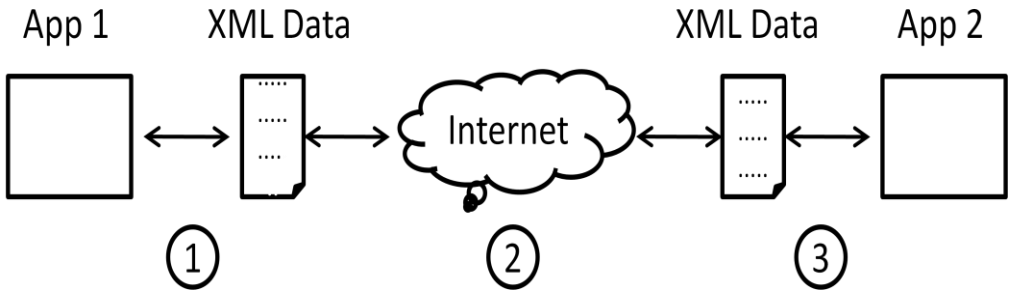

Figure 1: SOAP send/receive Data

From figure 1 above, one would depict that application 1 would use SOAP to generate the XML message (1), which is then send over the Internet (2), and application 2 would have to unmarshall the received XML message and read the data, then marshall any data again that will be send back to application 1 (3).

XML is used for representing data, just as HTML represents formatting of text to look like a web page. SOAP and HTTP allow these two languages to be put to use for delivering information. (NOTE: SOAP itself is sent over HTTP)

Definition

According to the W3C SOAP specification - SOAP is a lightweight protocol for exchange of information in a decentralized, distributed environment. It is an XML based protocol that consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined datatypes, and a convention for representing remote procedure calls and responses. SOAP can potentially be used in combination with a variety of other protocols; however, the common bindings is: SOAP in combination with HTTP and HTTP Extension Framework.

Analogy

Think of a SOAP message as an actual letter:

- The envelope: defines what the message is (an envelope in your mail box contains a letter)
- Header: to know this message is for (usually have the senders details as well)
- Letter: Encoded Message

Likewise, when an object receives a SOAP envelope, it knows to expect an encoded XML letter.

Skeleton of a SOAP Message:

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

<soap:Header>
  ...
  ...
</soap:Header>

<soap:Body>
  ...
  ...
  <soap:Fault>
    ...
    ...
  </soap:Fault>

</soap:Body>

</soap:Envelope>
```

The Envelope

The required SOAP Envelope element is the root element of a SOAP message. It defines the XML document as a SOAP message. Note the use of the xmlns:soap namespace, which points to: http://www.w3.org/2001/12/soap-envelope.

<SOAP: Body>: This element is part of the enveloped, and it is where the message is encoded.

The SOAP **encodingStyle** attribute is used to define the data types used in the document. This attribute may appear on any SOAP element, and it will apply to that element's contents and all child elements. A SOAP message has no default encoding.

Headers

There are two different types of headers SOAP uses for its messages. The first is a standard HTTP header that's also used for retrieving or sending HTML pages. This is appropriate as SOAP messages are also relayed via HTTP.

From Listing below:
POST /MyWebService HTTP/1.1
Host: MyHost
Content Type: text/xml; charset="utf-8"
Content-length: xxx

The HTTP header in the listing below is typical when using an HTTP-POST to send a SOAP message. The server would respond with a header as follows:

HTTP/1.1 200 OK
Content-Type: text/xml; charset=”utf-8”
Content-Length: xxx

```
POST /MyWebService HTTP/1.1
Host: MyHost
Content Type: text/xml; charset=”utf-8”
Content-length: xxx
…

<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

      <soap:Header>

        <m:Trans
        xmlns:m="http://www. example.com/transaction/"
        soap:mustUnderstand="1">234</m:Trans>

      </soap:Header>

...
...
</soap:Envelope>
```

The second type of header (useful to a developer), is known specifically as a SOAP header. This header is used to convey additional information that isn’t included in the body of the message or in the HTTP headers.</soap:Header>, It goes inside the envelope. The example above contains a header with a "Trans" element, a "mustUnderstand" attribute value of "1", and a value of 234.

SOAP defines three attributes in the default namespace ("http://www.w3.org/2001/12/soap-envelope") which defines how a recipient should process the SOAP message. These attributes are:

- actor,
- mustUnderstand, and
- encodingStyle.

The actor Attribute

A SOAP message may travel from a sender to a receiver by passing different endpoints along the message path. Not all parts of the SOAP message may be intended for the ultimate endpoint of the SOAP message but, instead, may be intended for one or more of the endpoints on the message path. The SOAP actor attribute may be used to address the Header element to a particular endpoint.

The mustUnderstand Attribute

The SOAP mustUnderstand attribute can be used to indicate whether a header entry is mandatory or optional for the recipient to process.

If you add "mustUnderstand="1" to a child element of the Header element it indicates that the receiver processing the Header must recognize the element. If the receiver does not recognize the element it must fail when processing the Header.

SOAP Body

The SOAP body contains the xml message intended for the ultimate end-point. The example below requests the price of apples. Note that the m:GetPrice and the Item elements above are application-specific elements. They are not a part of the SOAP standard.

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-
encoding">

<soap:Body>

   <m:GetPrice xmlns:m="http://www.exmaple.com/prices">
      <m:Item>Apples</m:Item>
   </m:GetPrice>

</soap:Body>

</soap:Envelope>
```

The SOAP response:

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-
encoding">
<soap:Body>

  <m:GetPriceResponse
   xmlns:m="http://www.example.com/prices">
      <m:Price>1.90</m:Price>
   </m:GetPriceResponse>

</soap:Body>
</soap:Envelope>
```

The SOAP Fault Element

An error message from a SOAP message is carried inside a Fault element.If a Fault element is present, it must appear as a child element of the Body element. A Fault element can only appear once in a SOAP message.

The SOAP Fault element has the following sub elements:

Sub Element	Description
<faultcode>	A code for identifying the fault
<faultstring>	A human readable explanation of the fault
<faultactor>	Information about who caused the fault to happen
<detail>	Holds application specific error information related to the Body element

SOAP Fault Codes

The faultcode values defined below must be used in the faultcode element when describing faults:

Error	Description
VersionMismatch	Found an invalid namespace for the SOAP Envelope element
MustUnderstand	An immediate child element of the Header element, with the mustUnderstand attribute set to "1", was not understood
Client	The message was incorrectly formed or contained incorrect information
Server	There was a problem with the server so the message could not proceed

SOAP HTTP Binding

A SOAP method is an HTTP request/response that complies with the SOAP encoding rules.

HTTP + XML = SOAP

A SOAP request could be an HTTP POST or an HTTP GET request. The HTTP POST request specifies at least two HTTP headers: Content-Type and Content-Length.

Content-Type

The Content-Type header for a SOAP request and response defines the MIME type for the message and the character encoding (optional) used for the XML body of the request or response.

Syntax :Content-Type: MIMEType; charset=character-encoding

Example:
POST /item HTTP/1.1
Content-Type: application/soap+xml; charset=utf-8

Content-Length

The Content-Length header for a SOAP request and response specifies the number of bytes in the body of the request or response

Syntax: Content-Length: bytes

Example:
POST /item HTTP/1.1
Content-Type: application/soap+xml; charset=utf-8
Content-Length: 250

Overview of a SOAP Engine

A SOAP engine (or processor) aids both consumers of Web services and their providers to accomplish their task without having to worry about the intricacies of SOAP message handling. As far as the consumer is concerned, it invokes an operation in a similar way a remote procedure call is invoked. The Web service provider needs to implement only the logic required by the business problem it solves. The consumer's SOAP processor converts the method invocation into a SOAP message. This message is transmitted through a transport, such as HTTP or SMTP, to the service provider's SOAP processor, which parses the message into a method invocation. The provider then executes the appropriate logic and gives the result to its SOAP processor, which parses the information into a SOAP response message. The message is transmitted through a transport to the consumer. It's SOAP processor parses the response message into a result object that it returns to the invoking entity.

Example: Axis is the third generation of Apache SOAP (an implementation of SOAP from the Apache Software Foundation). Axis is a SOAP engine as well as a code generator and WSDL processing tool. The Axis Web service processing model is shown below:

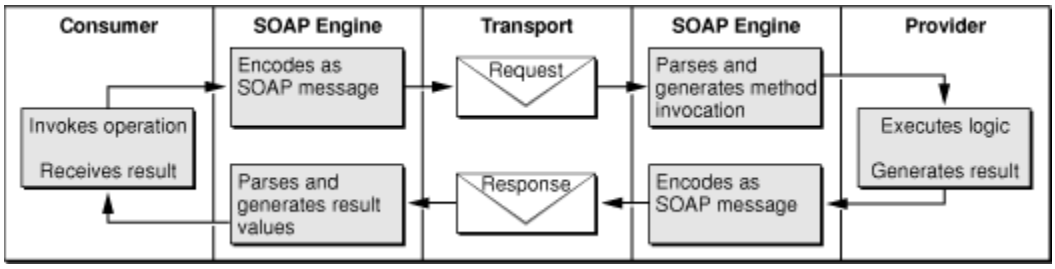

Figure 2: Web Service Processing Model