CS 683 Emerging Technologies
Spring Semester, 2003
Doc 12 SOAP

Contents

Web Services ................................................................. 2
SOAP Example – BabelFish Translation ............................. 4
SOAP Basic Structure..................................................... 10
  SOAP Envelope.......................................................... 11
  SOAP Headers........................................................... 14
SOAP Encoding ............................................................ 20
  Simple Types.............................................................. 21
  Multi-Referenced Value .............................................. 22
Enumerated Types .......................................................... 23
Compound Types ........................................................... 25

References

Simple Object Access Protocol (SOAP) 1.1
http://www.w3.org/TR/SOAP/

Numerous examples were copied from SOAP 1.1

Web Services Description Language (WSDL) 1.1
http://www.w3.org/TR/wsd1

X Methods, http://www.xmethods.net/, Bablefish Translation
Example

2003 SDSU & Roger Whitney, 5500 Campanile Drive, San Diego, CA 92182-7700 USA.
OpenContent (http://www.opencontent.org/opl.shtml) license defines the copyright on this
document.
Web Services

SOAP – Simple Object Access Protocol

WSDL – Web Services Description Language

UUDI – Universal Description, Discovery and Integration of Web Services
Some Background Reading

Unstalling SOAP, Dave Winer,  
http://davenet.userland.com/2001/03/29/unstallingSoap

• Some history about struggles behind the scenes over SOAP

2003 And Beyond, Andrew Grygus  
http://www.aaxnet.com/editor/edit029.html

• Long article with reference on current trends in technology

• A lot about Microsoft and .NET
SOAP Example – BabelFish Translation

Source: http://www.xmethods.net/

Overview of Service

Send:

• Text to be translated (this is a test)

• Source language (English – en)

• Target language (German – de)

Receive:

• Machine translation of text (dieses ist ein Test)

Basic Steps

• Request WSDL of service

• Use WSDL to formulate request

• Send request over http

• Receive result
Smalltalk Client Example

wsdlClient := WsdlClient url:
'http://www.xmethods.net/sd/BabelFishService.wsdl'.

message := Message selector: #BabelFish arguments: #('en_de' 'this is a test')

soapRequest := SoapRequest new.
soapRequest port: wsdlClient config anyPort.
soapRequest smalltalkEntity: message.

soapResponse := wsdlClient executeRequest: soapRequest.
The WSDL

About 1650 bytes

```
<definitions name="BabelFishService"
    targetNamespace="http://www.xmethods.net/sd/BabelFishService.wsdl"
    xmlns:tns="http://www.xmethods.net/sd/BabelFishService.wsdl"
    xmlns:xsd="http://www.w3.org/1999/XMLSchema"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
    <message name="BabelFishRequest">
        <part name="translationmode" type="xsd:string"/>
        <part name="sourcedata" type="xsd:string"/>
    </message>
    <message name="BabelFishResponse">
        <part name="return" type="xsd:string"/>
    </message>
    <portType name="BabelFishPortType">
        <operation name="BabelFish">
            <input message="tns:BabelFishRequest" name="BabelFish"/>
            <output message="tns:BabelFishResponse"
                name="BabelFishResponse"/>
        </operation>
    </portType>
```
WSDL Continued

<binding name="BabelFishBinding" type="tns:BabelFishPortType">
  <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="BabelFish">
    <soap:operation soapAction="urn:xmethodsBabelFish#BabelFish"/>
    <input>
      <soap:body use="encoded" namespace="urn:xmethodsBabelFish" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </input>
    <output>
      <soap:body use="encoded" namespace="urn:xmethodsBabelFish" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </output>
  </operation>
</binding>

<service name="BabelFish">
  <documentation>Translates text of up to 5k in length, between a variety of languages.</documentation>
  <port name="BabelFishPort" binding="tns:BabelFishBinding">
    <soap:address location="http://services.xmethods.net:80/perl/soaplite.cgi"/>
  </port>
</service>
The SOAP Request

POST /perl/soaplite.cgi HTTP/1.1
Host: services.xmethods.net
Content-length: 367
SOAPAction: "urn:xmethodsBabelFish#BabelFish"
Content-type: text/xml;charset=utf-8

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Body>
    <BabelFish
       SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
       xmlns="urn:xmethodsBabelFish">
      <translationmode>en_de</translationmode>
      <sourcedata>this is a test</sourcedata>
    </BabelFish>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
The SOAP Response

HTTP/1.1 200 OK
Date: Thr, 6 Mar 2003 01:39:58 -0800
Server: Apache/1.3.26 (Unix) Enhydra-Director/3 PHP/4.0.6 DAV/1.0.3
AuthNuSphere/1.0.0
Soapserver: SOAP::Lite/Perl/0.52
Content-length: 544
Keep-alive: timeout=15, max=100
Connection: Keep-Alive
Content-type: text/xml;charset=utf-8

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/1999/XMLSchema">
 <SOAP-ENV:Body>
  <namesp1:BabelFishResponse
 xmlns:namesp1="urn:xmethodsBabelFish">
   <return xsi:type="xsd:string">dieses ist ein Test</return>
  </namesp1:BabelFishResponse>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
SOAP Basic Structure

Basically oneway message sends

SOAP messages are in XML

Must not contain
  • DTD
  • Processing Instructions

Defines two XML namespaces

  • SOAP envelope http://schemas.xmlsoap.org/soap/envelope/
  • SOAP serialization http://schemas.xmlsoap.org/soap/encoding/

Parts of SOAP Message

  • Envelope (required)
  • Header (optional)
  • Body (required)
SOAP Envelope

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

Required

Envelope must be top-level tag

Envelope may contain

- Namespace declarations
- Additional attributes
- Sub-elements

Any attributes must be namespace-qualified

Sub-elements must
- Be Namespace-qualified
- Follow SOAP Body
Encoding Attribute

<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP-ENV:Body>

States rules used to serialize/deserialize soap message

Standard value is

http://schemas.xmlsoap.org/soap/encoding/

Can list of URIs

"http://my.host/encoding/restricted http://my.host/encoding/"

Empty URI "" indicates no claims are made about encoding style
Versions

<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"

Envelope namespace is version of SOAP being used

Text claims this is a good thing

Spec states:

MUST have an Envelope element associated with the
"http://schemas.xmlsoap.org/soap/envelope/ " namespace

If a message is received by a SOAP application in which the
SOAP Envelope element is associated with a different
namespace, the application MUST treat this as a version
error and discard the message.

Text tries to explain how SOAP engine can use URI as versions


SOAP Headers

```xml
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
 <SOAP-ENV:Header>
   <t:Transaction
     xmlns:t="some-URI"
     SOAP-ENV:mustUnderstand="1"> 5
   </t:Transaction>
 </SOAP-ENV:Header>
 <SOAP-ENV:Body>
   <m:GetLastTradePrice xmlns:m="Some-URI">
     <symbol>DEF</symbol>
   </m:GetLastTradePrice>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Used to provide additional information about message

If exists must be first element in Envelope

Header entries must be namespace qualified

**Special Attributes**

- encodingStyle
- mustUnderstand
- actor
mustUnderstand

Possible values:

1, 0

If mustUnderstand is not present is has value 0

Value 1 means recipient of that header entry MUST
• obey the semantics and process correctly or
• fail processing the message (return an error)
actor

SOAP messages may pass through multiple SOAP processors

Actor attribute specifies parts of the message to be for specific processors

SOAP processor does not forward headers meant for it

If actor attribute is omitted the recipient is the ultimate destination of the message

"http://schemas.xmlsoap.org/soap/actor/next " indicates header is for the first SOAP processor
SOAP Faults

Used to indicate an error in processing a SOAP message

If present
• Must be a body entry
• Cannot appear more than once

Subelements

faultcode
  Code to identify the error

faultstring
  Human readable explanation of the error

faultactor
  Information about which actor the error occurred

detail
  Application specific information about error
Error Codes

VersionMismatch
   Invalid namespace for SOAP Envelope

MustUnderstand
   Immediate child element of SOAP Header was not understood

Client
   Messages was incorrectly formed

Server
   Server could not process message

Each error can be extended with “.” dot notation

Client.Authentication
Fault Example

HTTP/1.1 500 Internal Server Error
Date: Thr, 6 Mar 2003 17:46:23 -0800
Server: Apache/1.3.26 (Unix) Enhydra-Director/3 PHP/4.0.6
DAV/1.0.3 AuthNuSphere/1.0.0
Soapserver: SOAP::Lite/Perl/0.52
Content-length: 700
Connection: close
Content-type: text/xml;charset=utf-8

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
    xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/
    SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/
    xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/1999/XMLSchema-instance">
    <SOAP-ENV:Body>
        <SOAP-ENV:Fault>
            <faultcode xsi:type="xsd:string">SOAP-ENV:Server</faultcode>
            <faultstring xsi:type="xsd:string">Could not translate. Either the service cannot handle your input, your connection to the babelfish site may be down or the site itself may be expiencing difficulties</faultstring>
        </SOAP-ENV:Fault>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
SOAP Encoding

SOAP supports

• Simple types
• Compound types
• Multi-dimensional arrays
• Sparse Arrays
• Structures
• Circular references to types
Simple Types

Uses built-in data types of XML Schema

Built-in Datatype Hierarchy

- `anyType`
- `anySimpleType`
- `all complex types`

- `duration`
- `dateTime`
- `time`
- `date`
- `gYearMonth`
- `gYear`
- `gMonthDay`
- `gDay`
- `gMonth`
- `boolean`
- `base64Binary`
- `hexBinary`
- `float`
- `double`
- `anyURI`
- `QName`
- `NOTATION`

- `string`
- `normalizedString`
- `decimal`
- `integer`

- `token`
- `nonPositiveInteger`
- `long`
- `nonNegativeInteger`

- `language`
- `Name`
- `NMTOKEN`
- `negativeInteger`
- `int`
- `unsignedLong`
- `positiveInteger`

- `NCName`
- `NMTOKENS`
- `short`
- `unsignedInt`

- `ID`
- `IDREF`
- `ENTITY`
- `IDREFS`
- `ENTITIES`
- `byte`
- `unsignedShort`
- `unsignedByte`

- `ur types`
- `built-in primitive types`
- `built-in derived types`
- `complex types`

- derived by restriction
- derived by list
- derived by extension or restriction
Multi-Referenced Value

Two Tags, Different String instances

<greeting>Hello</greeting>
<salutation>Hello</salutation>

Two Tags, One String instance

<greeting id="String-0">Hello</greeting>
<salutation href="#String-0"/>
Enumerated Types

Tags in SOAP Message

<Person>
  <Name>Henry Ford</Name>
  <Age>32</Age>
  <EyeColor>Brown</EyeColor>
</Person>

Schema in WSDL

<element name="EyeColor" type="tns:EyeColor"/>
<simpleType name="EyeColor" base="xsd:string">
  <enumeration value="Green"/>
  <enumeration value="Blue"/>
  <enumeration value="Brown"/>
</simpleType>
Array of Bytes

<picture xsi:type="SOAP-ENC:base64">
  aG93IG5vDyBicm73biBjb3cNCg==
</picture>
Compound Types

Struct
   Each value has a name

Array
   Ordinal position indicated individual values
Struct

Simple Example

<e:Book>
  <author>Henry Ford</author>
  <preface>Prefatory text</preface>
  <intro>This is a book.</intro>
</e:Book>

References

<e:Book>
  <title>My Life and Work</title>
  <author href="#Person-1"/>
</e:Book>
<e:Person id="Person-1">
  <name>Henry Ford</name>
  <address href="#Address-2"/>
</e:Person>
<e:Address id="Address-2">
  <email>mailto:henryford@hotmail.com</email>
  <web>http://www.henryford.com</web>
</e:Address>

External References

<e:Book>
  <title>Paradise Lost</title>
  <firstauthor href="http://www.dartmouth.edu/~milton"/>
</e:Book>
Arrays

Sample WDSL Schema

<element name="myFavoriteNumbers"
    type="SOAP-ENC:Array"/>

Array with declared Element Tags

<myFavoriteNumbers
    SOAP-ENC:arrayType="xsd:int[2]">
    <number>3</number>
    <number>4</number>
</myFavoriteNumbers>

Array with xsi:attribute

<SOAP-ENC:Array
    SOAP-ENC:arrayType="xsd:int[2]">
    <SOAP-ENC:int>3</SOAP-ENC:int>
    <SOAP-ENC:int>4</SOAP-ENC:int>
</SOAP-ENC:Array>
Different Types

<SOAP-ENC:Array SOAP-ENC:arrayType="xsd:ur-type[4]">
  <SOAP-ENC:int>12345</SOAP-ENC:int>
  <SOAP-ENC:decimal>6.789</SOAP-ENC:decimal>
  <xsd:string>
    Of Mans First Disobedience, and the Fruit
    Of that Forbidden Tree, whose mortal tast
    Brought Death into the World, and all our woe,
  </xsd:string>
  <SOAP-ENC:uriReference>
    http://www.dartmouth.edu/~milton/reading_room/
  </SOAP-ENC:uriReference>
</SOAP-ENC:Array>
Array Subtypes

WSDL Schema

<simpleType name="phoneNumber" base="string"/>

<element name="ArrayOfPhoneNumbers">
  <complexType base="SOAP-ENC:Array">
    <element name="phoneNumber" type="tns:phoneNumber" maxOccurs="unbounded"/>
  </complexType>
  <anyAttribute/>
</element>

Sample Array

xyz:ArrayOfPhoneNumbers SOAP-ENC:arrayType="xyz:phoneNumber[2]">
  <phoneNumber>206-555-1212</phoneNumber>
  <phoneNumber>1-888-123-4567</phoneNumber>
</xyz:ArrayOfPhoneNumbers>
Multi-dimensional Array

<SOAP-ENC:Array SOAP-ENC:arrayType="xsd:string[2,3]">  
  <item>r1c1</item>  
  <item>r1c2</item>  
  <item>r1c3</item>  
  <item>r2c1</item>  
  <item>r2c2</item>  
  <item>r2c3</item>  
</SOAP-ENC:Array>

Partially Transmitted Array

<SOAP-ENC:Array SOAP-ENC:arrayType="xsd:string[5]" SOAP-ENC:offset="[2]">  
  <item>The third element</item>  
  <item>The fourth element</item>  
</SOAP-ENC:Array>

Sparse Array

<SOAP-ENC:Array SOAP-ENC:arrayType="xsd:string[,][4]">  
  <SOAP-ENC:Array href="#array-1" SOAP-ENC:position="[2]"/>  
</SOAP-ENC:Array>

<SOAP-ENC:Array id="array-1" SOAP-ENC:arrayType="xsd:string[10,10]">  
  <item SOAP-ENC:position="[2,2]">Third row, third col</item>  
  <item SOAP-ENC:position="[7,2]">Eighth row, third col</item>  
</SOAP-ENC:Array>