CS 683 Emerging Technologies
Spring Semester, 2003
Doc 16 SOAP Server 2
Contents

References

Axis documentation and examples, http://ws.apache.org/axis/

Axis

Deploying Web Service with WSDD

User guide example 3 from Axis Distribution

MyService.java

```
public class MyService
{
   public String serviceMethod(String arg)
   {
      return arg;
   }
}
```

deploy.wsdd

```
<deployment xmlns="http://xml.apache.org/axis/wsdd/"
          xmlns:java="http://xml.apache.org/axis/wsdd/providers/java">

 <service name="MyService" provider="java:RPC">
  <parameter name="className" value="MyService"/>
  <parameter name="allowedMethods" value="*"/>
 </service>

</deployment>
```

Running the Example

Compile

Compile MyService.java

Place class files in TomcatDirectory/shared/classes

or

Place jar file in TomcatDirectory/shared/lib

Register service

Use the admin client

Assuming axis.jar is in your path, in the directory containing your wsdd file

java org.apache.axis.client.AdminClient deploy.wsdd

Response:

- Processing file deploy.wsdd
- <Admin>Done processing</Admin>

Client.java

```
import org.apache.axis.client.Call;
import org.apache.axis.client.Service;
import org.apache.axis.encoding.XMLType;
import org.apache.axis.utils.Options;
import javax.xml.namespace.QName;
import javax.xml.rpc.ParameterMode;

public class Client
{
   public static void main(String [] args)
   {
      try {
         Options options = new Options(args);

         String endpointURL = options.getURL();
         String textToSend;

         args = options.getRemainingArgs();
         if ((args == null) || (args.length < 1)) {
            textToSend = "<nothing>";
         } else {
            textToSend = args[0];
         }

         Service  service = new Service();
         Call  call = (Call) service.createCall();

         call.setTargetEndpointAddress( new java.net.URL(endpointURL) );
         call.setOperationName( "serviceMethod" );
         call.addParameter( "arg1",
               XMLType.XSD_STRING, ParameterMode.IN);
         call.setReturnType( org.apache.axis.encoding.XMLType.XSD_STRING
);
```

```
            String ret = (String) call.invoke( new Object[] { textToSend } );

            System.out.println("You typed : " + ret);
        } catch (Exception e) {
            System.err.println(e.toString());
        }
    }
}
```

Running the Client

java Client -lhttp://localhost:8080/axis/services/MyService "cat”

Unregistering the Service

undeploy.wsdd

```
<undeployment xmlns="http://xml.apache.org/axis/wsdd/">
 <service name="MyService"/>
</undeployment>
```

The command

```
java org.apache.axis.client.AdminClient deploy.wsdd
```

Axis Service Styles

- RPC
- Document
- Wrapped
- Message

RPC

Axis maps request to a Java method

XML data is mapped to predefined Java types

Document

Requests are mapped to Java Method

Data is mapped to a “Struct” class with fields for each of the XML types sent

Wrapped

Requests mapped to Java Method

Java method name is top level type of XML data sent

Arguments are second level XML data sent

Example

Soap

```
<soap:Envelope xmlns="http://xml.apache.org/axis/wsdd/"
[redacted]xmlns:java="http://xml.apache.org/axis/wsdd/providers/java">
  <soap:Body>
    <myNS:PurchaseOrder xmlns:myNS="http://commerce.com/PO">
      <item>SK001</item>
      <quantity>1</quantity>
      <description>Sushi Knife</description>
    </myNS:PurchaseOrder>
  </soap:Body>
</soap:Envelope>
```

Schema

```
<schema targetNamespace="http://commerce.com/PO">
  <complexType name="POType">
    <sequence>
      <element name="item" type="xsd:string"/>
      <element name="quantity" type="xsd:int"/>
      <element name="description" type="xsd:string"/>
    </sequence>
  </complexType>
  <element name="PurchaseOrder" type="POType"/>
</deployment>
```

Method called

```
public void purchaseOrder(String item, int quantity,
      String    description)
```

Message

Axis passes XML to your Java method

Possible methods

- public Element [] method(Element [] bodies)

Elements contains DOM Element for each XML element in SOAP request

- public SOAPBodyElement [] method (SOAPBodyElement [] bodies)

SOAPBodyElement contains SOAPBodyElement for each XML element in SOAP request

- public Document method(Document body)

body is a DOM Document representing SOAP request

- public void method(SOAPEnvelope req, SOAPEnvelope resp)

Get the SOAP request and response objects

Mapping between Java data types and XML

See Java API for XML-Based RPC for all the details
http://java.sun.com/xml/jaxrpc/

Basic Types Mapped Automatically

Simple Type	Java Type
xsd:string	java.lang.String
xsd:integer	java.math.BigInteger
xsd:int	int
xsd:long	long
xsd:short	short
xsd:decimal	java.math.BigDecimal
xsd:float	float
xsd:double	double
xsd:boolean	boolean
xsd:byte	byte
xsd:Qname	javax.xml.namespace.QName
xsd:dateTime	java.util.Calendar
xsd:base64Binary	byte[]
xsd:hexBinary	byte[]

Exceptions

java.rmi.RemoteException and subclasses map to SOAP Fault

Fault code is name of exception class

Other exceptions map to wsdl:fault

Using Default Bean Mappers

A Bean is a class with getX() setX() methods for each field

From Axis samples/userguid/example5
Author Glen Daniels


```
public class Order
{
   private String customerName;
   private String shippingAddress;
   private String itemCodes[];
   private int quantities[];

   public String getCustomerName() { return customerName; }
   public void setCustomerName(String name)
     { customerName = name; }

   public String getShippingAddress() { return shippingAddress; }
   public void setShippingAddress(String address)
   { shippingAddress = address; }

   public String [] getItemCodes() { return itemCodes; }
   public void setItemCodes(String [] items)
   { itemCodes = items; }

   public int [] getQuantities()  { return quantities; }
   public void setQuantities(int [] quants)
   { quantities = quants; }
}
```

Specify in WSDD the Bean

```
<deployment xmlns="http://xml.apache.org/axis/wsdd/"
        xmlns:java="http://xml.apache.org/axis/wsdd/providers/java">

 <service name="OrderProcessor" provider="java:RPC">
  <parameter name="className"
     value="samples.userguide.example5.BeanService"/>
  <parameter name="allowedMethods" value="processOrder"/>

  <beanMapping qname="myNS:Order"
     xmlns:myNS="urn:BeanService"


   languageSpecificType="java:samples.userguide.example5.Order"/>
 </service>

</deployment>
```

The Service

```
package samples.userguide.example5;

public class BeanService
{
   public String processOrder(Order order)
   {
      String sep = System.getProperty("line.separator");

      String response = "Hi, " + order.getCustomerName() + "!" + sep;

      response += sep + "You seem to have ordered the following:" + sep;

      String [] items = order.getItemCodes();
      int [] quantities = order.getQuantities();

      for (int i = 0; i < items.length; i++) {
         response += sep + quantities[i] + " of item : " + items[i];
      }

      response += sep + sep +
               "If this had been a real order processing system, "+
               "we'd probably have charged you about now.";

      return response;
   }
}
```

Custom Mappings

You need:

- Factories
- Serializers/Deserializers
- WSDD entry

Example – Axis samples/encoding

Data.java

```
package samples.encoding;

public class Data {
   public String stringMember;
   public Float floatMember;
   public Data dataMember;

   public String toString()  {
      return getStringVal("", this);
   }

   public String getStringVal(String indent, Data topLevel)   {
      String ret = "\n" + indent + "Data:\n";
      ret +=       indent + " str[" + stringMember + "]\n";
      ret +=       indent + " float[" + floatMember + "]\n";
      ret +=       indent + " data[";

      if (dataMember != null) {
         if (dataMember == topLevel) {
            ret += " top level";
         } else
            ret += dataMember.getStringVal(indent + " ", topLevel)+"\n" +
indent;
      } else
         ret += " null";

      ret += " ]";
      return ret;
   }
}
```

Factories

DeserializerFactory

```
package samples.encoding;
import org.xml.sax.Attributes;
import org.xml.sax.SAXException;
import javax.xml.namespace.QName;
import java.io.IOException;
import org.apache.axis.encoding.Serializer;
import org.apache.axis.encoding.SerializerFactory;
import org.apache.axis.encoding.SerializationContext;
import org.apache.axis.encoding.Deserializer;
import org.apache.axis.encoding.DeserializerFactory;
import org.apache.axis.encoding.DeserializationContext;
import org.apache.axis.encoding.Deserializer;
import org.apache.axis.Constants;

import java.util.Iterator;
import java.util.Vector;

public class DataDeserFactory implements DeserializerFactory {
   private Vector mechanisms;

   public DataDeserFactory() {
   }
   public javax.xml.rpc.encoding.Deserializer getDeserializerAs(
         String mechanismType) {
      return new DataDeser();
   }
   public Iterator getSupportedMechanismTypes() {
      if (mechanisms == null) {
         mechanisms = new Vector();
         mechanisms.add(Constants.AXIS_SAX);
      }
      return mechanisms.iterator();
   }
}
```

```
package samples.encoding;

import org.xml.sax.Attributes;
import org.xml.sax.SAXException;
import javax.xml.namespace.QName;
import java.io.IOException;
import org.apache.axis.encoding.Serializer;
import org.apache.axis.encoding.SerializerFactory;
import org.apache.axis.encoding.SerializationContext;
import org.apache.axis.encoding.Deserializer;
import org.apache.axis.encoding.DeserializerFactory;
import org.apache.axis.encoding.DeserializationContext;
import org.apache.axis.encoding.Deserializer;
import org.apache.axis.Constants;

import java.util.Iterator;
import java.util.Vector;

public class DataSerFactory implements SerializerFactory {
   private Vector mechanisms;

   public DataSerFactory() {
   }
   public javax.xml.rpc.encoding.Serializer getSerializerAs(String
         mechanismType) {
      return new DataSer();
   }
   public Iterator getSupportedMechanismTypes() {
      if (mechanisms == null) {
         mechanisms = new Vector();
         mechanisms.add(Constants.AXIS_SAX);
      }
      return mechanisms.iterator();
   }
}
```

Serializer

```
package samples.encoding;

import org.apache.axis.encoding.SerializationContext;
import org.apache.axis.encoding.Serializer;
import org.apache.axis.message.SOAPHandler;
import org.apache.axis.Constants;
import org.xml.sax.Attributes;
import org.xml.sax.SAXException;
import org.apache.axis.Constants;
import org.apache.axis.wsdl.fromJava.Types;
import javax.xml.namespace.QName;
import java.io.IOException;
import java.util.Hashtable;

public class DataSer implements Serializer
{
   public static final String STRINGMEMBER = "stringMember";
   public static final String FLOATMEMBER = "floatMember";
   public static final String DATAMEMBER = "dataMember";
   public static final QName myTypeQName = new QName("typeNS",
"Data");
```

```
  public void serialize(QName name, Attributes attributes,
                        Object value, SerializationContext context)
      throws IOException
  {
      if (!(value instanceof Data))
          throw new IOException("Can't serialize a " +
value.getClass().getName()
        + " with a DataSerializer.");
      Data data = (Data)value;

      context.startElement(name, attributes);
      context.serialize(
        new QName("", STRINGMEMBER), null, data.stringMember);

      context.serialize(
           new QName("", FLOATMEMBER), null, data.floatMember);

      context.serialize(
           new QName("", DATAMEMBER), null, data.dataMember);

      context.endElement();
  }

  public String getMechanismType() { return Constants.AXIS_SAX; }

  public boolean writeSchema(Types types) throws Exception {
      return false;
  }
}
```

Deserializer

```
package samples.encoding;

import org.apache.axis.encoding.DeserializationContext;
import org.apache.axis.encoding.Deserializer;
import org.apache.axis.encoding.DeserializerImpl;
import org.apache.axis.encoding.FieldTarget;
import org.apache.axis.Constants;
import org.apache.axis.message.SOAPHandler;
import org.xml.sax.Attributes;
import org.xml.sax.SAXException;

import javax.xml.namespace.QName;
import java.util.Hashtable;

public class DataDeser extends DeserializerImpl
{
   public static final String STRINGMEMBER = "stringMember";
   public static final String FLOATMEMBER = "floatMember";
   public static final String DATAMEMBER = "dataMember";
   public static final QName myTypeQName = new QName("typeNS",
"Data");

   private Hashtable typesByMemberName = new Hashtable();

   public DataDeser()
   {
      typesByMemberName.put(STRINGMEMBER,
Constants.XSD_STRING);
      typesByMemberName.put(FLOATMEMBER, Constants.XSD_FLOAT);
      typesByMemberName.put(DATAMEMBER, myTypeQName);
      value = new Data();
   }
```

```
    /** DESERIALIZER STUFF - event handlers
     */

    public SOAPHandler onStartChild(String namespace,
                                    String localName,
                                    String prefix,
                                    Attributes attributes,
                                    DeserializationContext context)
        throws SAXException
    {
        QName typeQName = (QName)typesByMemberName.get(localName);
        if (typeQName == null)
            throw new SAXException("Invalid element in Data struct - " +
                 localName);

        // These can come in either order.
        Deserializer dSer = context.getDeserializerForType(typeQName);
        try {

            dSer.registerValueTarget(new FieldTarget(value, localName));

        } catch (NoSuchFieldException e) {
            throw new SAXException(e);
        }

        if (dSer == null)
            throw new SAXException("No deserializer for a " + typeQName );

        return (SOAPHandler)dSer;
    }
}
```

WSDD

```
<deployment name="test"
   xmlns="http://xml.apache.org/axis/wsdd/"

   xmlns:java="http://xml.apache.org/axis/wsdd/providers/java"
   xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance">
  <service name="ElementService" provider="java:RPC">
    <parameter name="className"
         value="samples.encoding.ElementService" />
    <parameter name="allowedMethods" value="echoElement" />
   <typeMapping qname="typeNS:Data"
      xmlns:typeNS="urn:DataExample"
      languageSpecificType="java:samples.encoding"
      serializer="samples.encoding..DataSerFactory"
      deserializer="samples.encoding.DataDeserFactory"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
  </service>
</deployment>
```