

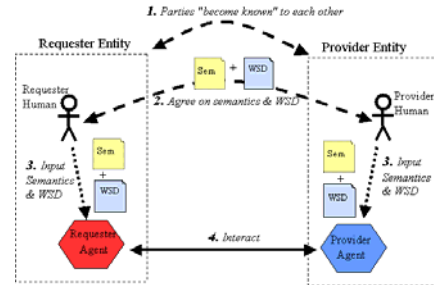
Web Services

- The World Wide Web is increasingly used for application-to-application communication.
- The programmatic interfaces made available are referred to as **Web services**.
- Definition: A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (mainly **WSDL**). Other systems interact with the Web service in a manner prescribed by its description using **SOAP** messages, typically conveyed using **HTTP** with **XML serialization** in conjunction with other Web-related standards

[W3C Web Services Working Group]

1

Web Service Interaction



2

SOAP

- SOAP is an XML-based communication protocol and encoding format for inter-application communication.
 - Communication between programs written in arbitrary languages, across the Internet.
- Wire protocol
 - Structure of exchange messages between SOAP systems
- Originally conceived by Microsoft and Userland software
- The W3C's XML Protocol working group is in charge of the specification.
- Modes:
 - Remote Procedure Call (RPC)
 - Similar to method invocation, but on remote machine
 - Exchanges messages over HTTP: the client POSTs a SOAP request, and receives either an HTTP success code and a SOAP response or an HTTP error code.
 - One-way messaging
 - SOAP implementations over SMTP

3

SOAP Envelope Element

- Top-level XML element of a SOAP request or response
- Must be present
- Includes
 - Namespace declarations
 - Encoding style
- May contain a Header element
- Must contain a Body element

4

SOAP Header Element

- Information for authentication, processing or routing request

```
<soapenv:Header>
  <ns1:username xmlns:ns1="JavaSoapBook">
    Jessica
  </ns1:username>
</soapenv:Header>
```

5

SOAP Body Element

- Encode information that represents RPC
 - Request
 - Method name
 - Parameters
 - Response
 - Return value

6

Sample Web Service

```
public class Calculator {
    public int add(int i1, int i2)
    {
        return i1 + i2;
    }

    public int subtract(int i1, int i2)
    {
        return i1 - i2;
    }
}
```

7

SOAP Request

```
POST /axis/Calculator.jws HTTP/1.0
Host: localhost
Content-Type: text/xml
Content-Length: 586
SOAPAction: ""

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <calc:add xmlns:calc="Calculator">
      <calc:i1 xsi:type="xsd:int">50</calc:i1>
      <calc:i2 xsi:type="xsd:int">6</calc:i2>
    </calc:add>
  </soapenv:Body>
</soapenv:Envelope>
```

8

SOAP Response

```
HTTP/1.1 200 OK
Set-Cookie: JSESSIONID=7B0EBF2F233CED6D33E6E4D172BC0631; Path=/axis
Content-Type: text/xml;charset=utf-8
Date: Tue, 23 Nov 2004 03:42:37 GMT
Server: Apache-Coyote/1.1
Connection: close

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" >
  <soapenv:Body>
    <ns1:addResponse xmlns:ns1="Calculator">
      <addReturn xsi:type="xsd:int" />56</addReturn>
    </ns1:addResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

9

SOAP Error

```
HTTP/1.1 500 Internal Server Error
Content-Type: text/xml;charset=utf-8
Date: Tue, 23 Nov 2004 03:59:35 GMT
Server: Apache-Coyote/1.1
Connection: close

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <soapenv:Fault>
      <faultcode>soapenv:Server.userException</faultcode>
      <faultstring>java.io.FileNotFoundException: /Calculato.jws</faultstring>
      <detail>
        <ns1:hostname xmlns:ns1="http://xml.apache.org/axis/">simon</ns1:hostname>
      </detail>
    </soapenv:Fault>
  </soapenv:Body>
</soapenv:Envelope>
```

10

Hand-Coded SOAP Example

11

Web Services Description Language (WSDL)

- XML grammar for describing Web Services
 - Service location
 - Service methods
 - Parameters and return types
 - Data types
- Specification authored by IBM and Microsoft
- <http://www.w3.org/TR/wsdl>

12

Stock Quote Example

```
public interface StockQuote {
    public QuoteData getQuote(String symbol, String LicenseKey);
}

public class QuoteData implements java.io.Serializable {
    String stockSymbol;
    BigDecimal lastTradeAmount;
    Calendar lastTradeDateTime;
    BigDecimal stockChange;
    BigDecimal openAmount;
    BigDecimal dayHigh;
    BigDecimal dayLow;
    int stockVolume;
    String mktCap;
    BigDecimal prevCls;
    String changePercent;
    String fiftyTwoWeekRange;
    .....
}
```

13

WSDL Document

- Collection of one or more service definitions

```
<definitions>
  <types> ..... </types>
  <message> .... </message>
  <portType>
    <operation> .... </operation>
    .....
  </portType>
  <binding>
    <operation> .... </operation>
    .....
  </binding>
  <service> ..... </service>
</definitions>
```

14

definitions

- Root of the document
- Includes name space definitions
- Target name space for service definitions

```
<definitions
  xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:s="http://www.w3.org/2001/XMLSchema"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding"
  xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
  .....
  targetNamespace="http://ws.cdyne.com/" >
```

15

types

- A container for data type definitions based on XML Schema
 - Custom types used by service

```
<types>
  <schema elementFormDefault="qualified" targetNamespace="http://ws.cdyne.com/">
    <import namespace="http://www.w3.org/2001/XMLSchema" />
    <complexType name="QuoteData">
      <sequence>
        <element minOccurs="0" maxOccurs="1" name="StockSymbol" type="s:string" />
        <element minOccurs="1" maxOccurs="1" name="LastTradeAmount" type="s:decimal" />
        <element minOccurs="1" maxOccurs="1" name="LastTradeDateTime" type="s:dateTime" />
        <element minOccurs="1" maxOccurs="1" name="StockChange" type="s:decimal" />
        <element minOccurs="1" maxOccurs="1" name="OpenAmount" type="s:decimal" />
        <element minOccurs="1" maxOccurs="1" name="DayHigh" type="s:decimal" />
        <element minOccurs="1" maxOccurs="1" name="DayLow" type="s:decimal" />
        <element minOccurs="1" maxOccurs="1" name="StockVolume" type="s:int" />
        <element minOccurs="0" maxOccurs="1" name="MktCap" type="s:string" />
        <element minOccurs="1" maxOccurs="1" name="PrevCls" type="s:decimal" />
        <element minOccurs="0" maxOccurs="1" name="ChangePercent" type="s:string" />
        <element minOccurs="0" maxOccurs="1" name="CompanyName" type="s:string" />
        .....
      </sequence>
    </complexType>
```

16

types (cont.)

```
<s:element name="GetQuote">
  <s:complexType>
    <s:sequence>
      <s:element name="StockSymbol" type="s:string" />
      <s:element name="LicenseKey" type="s:string" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="GetQuoteResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetQuoteResult" type="s0:QuoteData" />
    </s:sequence>
  </s:complexType>
</s:element>
</s:schema>
</types>
```

17

message

- An abstract, typed definition of the data being communicated

```
<message name="GetQuoteIn">
  <part name="parameters" element="s0:GetQuote" />
</message>
<message name="GetQuoteOut">
  <part name="parameters" element="s0:GetQuoteResponse" />
</message>
```

18

portType

- An abstract set of operations supported by the service
- <operation>
 - An abstract description of an action supported by the service

```
<portType name="StockQuote">
  <operation name="GetQuote">
    <documentation>
      This method retrieves a current stock quote. Use a license
      key of 0 for testing.
    </documentation>
    <input message="s0:GetQuoteIn" />
    <output message="s0:GetQuoteOut" />
  </operation>
</portType>
```

19

binding

- A concrete protocol and data format specification for a particular portType

```
<binding name="StockQuoteBind" type="s0:StockQuote">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
    style="document" />
  <operation name="GetQuote">
    <soap:operation soapAction="http://ws.cdyne.com/GetQuote"
      style="document" />
    <input><soap:body use="literal" /></input>
    <output><soap:body use="literal" /></output>
  </operation>
</binding>
```

20

service

- port: A single endpoint defined as a combination of a binding and a network address
- service: A collection of related endpoints

```
<service name="Stocks">
  <port name="StockQuotePort" binding="s0:StockQuoteBind">
    <soap:address
      location="http://ws.cdyne.com/delayedstockquote/delayedstockquote.asmx" />
  </port>
</service>
```

21

Apache AXIS

- Axis is essentially a SOAP engine -- a framework for constructing SOAP processors such as clients, servers, gateways, etc.
 - New incarnation of Apache SOAP (which began at IBM as "SOAP4J")
- <http://ws.apache.org/axis/>
- Axis handles the "magic" of converting Java objects to SOAP data when it sends it over the wire or receives results.
- SOAP Faults are sent by the server when something goes wrong; Axis converts these to Java exceptions.
- Axis is written in Java
 - Compiled in the JAR file axis.jar

22

AXIS Client Side

- Allows you to easily build stubs to access remote service.
- "WSDL2Java" builds Java classes (stubs) from WSDL descriptions.

23

WSDL2Java

```
java org.apache.axis.wsdl.WSDL2Java FileName.wsdl
```

- Generates client-side bindings.
 - Follows the JAX-RPC specification

WSDL clause	Java class(es) generated
For each type or element	A java class
For each portType	A java interface
For each binding	A stub class
For each service	A service interface
	A service implementation (the locator)

24

WSDL2Java StockQuote

```
java org.apache.axis.wsdl.WSDL2Java StockQuote.wsdl
```

Generates

- Interfaces
 - StockQuote
 - Stocks
- Classes
 - QuoteData
 - StockQuoteBindingStub
 - StockLocator

25

WSDL2Java-based StockClient

26

AXIS Server Side

- Can export a Java class as a web service
- A simple stand-alone server,
- A server which plugs into servlet engines such as Tomcat
- Tool that generates WSDL from Java classes

27

Axis and Tomcat

- Axis is a Web Application (group of Java Servlets)
- Installed at:
 - \$CATALINA_HOME/webapps/axis
- Java classes that implement a web services are installed at:
 - \$CATALINA_HOME/webapps/axis/WEB_INF/classes
- Axis configuration file:
 - \$CATALINA_HOME/webapps/axis/WEB_INF/server-config.wsdd

28

Sample Web Service

```
public class Calculator {
    public int add(int i1, int i2)
    {
        return i1 + i2;
    }

    public int subtract(int i1, int i2)
    {
        return i1 - i2;
    }
}
```

29

Deploying Axis Applications

- 1) Add new compiled classes to the Axis webapp
 - Place Calculator.class to \$CATALINA_HOME/webapps/axis/WEB_INF/classes
- 2) Register Web service
 - **Axis Web Service Deployment Descriptor (WSDD)** format

```
<deployment xmlns="http://xml.apache.org/axis/wsdd/"
  xmlns:java="http://xml.apache.org/axis/wsdd/providers/java">
  <service name="Calculator" provider="java:RPC">
    <parameter name="className" value="Calculator"/>
    <parameter name="allowedMethods" value="*/"/>
    <parameter name="scope" value="Application"/>
  </service>
</deployment>
```
 - Axis AdminClient

```
java org.apache.axis.client.AdminClient deploy.wsdd
```
 - Assumes: - CLASSPATH includes axis.jar, jaxrpc.jar, saaj.jar, commons-discovery.jar, and commons-logging.jar
- Tomcat is running

30

Service Scope

- Axis supports scoping service objects (the actual Java objects which implement your methods) three ways.
 - "Request" scope, the default, will create a new object each time a SOAP request comes in for your service.
 - "Application" scope will create a single shared object to service **all** requests.
 - "Session" scope will create a new object for each session-enabled client who accesses your service.

```
<parameter name="scope" value="Application"/>
```

31

Java2WSDL

- Creates wsdl description from Java interface or class

```
public class Calculator {
    public int add(int i1, int i2) {
        return i1 + i2;
    }

    public int subtract(int i1, int i2) {
        return i1 - i2;
    }
}

java org.apache.axis.wsdl.Java2WSDL
-o calculator.wsdl // wsdl file name
-l "http://localhost:8080/axis/services/Calculator" // location of service
-n "urn:Calculator" // namespace of WSDL file
Calculator
```

32

WSDL for Deployed Axis Services

- When you deploy a service in Axis, users may then access your service's URL with a standard web browser and by appending "?WSDL" to the end of the URL, they will obtain an automatically-generated WSDL document which describes your service.

33

WSDL2Java Calculator

```
java org.apache.axis.wsdl.WSDL2Java calculator.wsdl
```

Generates:

- Package Calculator_pkg
- Interfaces
 - Calculator
 - CalculatorService
- Classes
 - CalculatorBindingStub
 - CalculatorServiceLocator

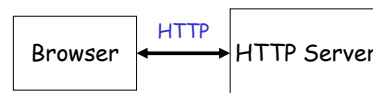
34

WSDL2Java-based CalculatorClient

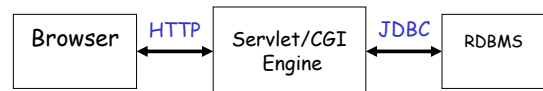
35

Web Application Architecture

- Static content



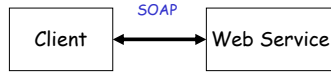
- Dynamic content



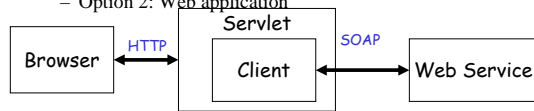
36

Web Application Architecture

- Web Service
 - Option 1: Native application



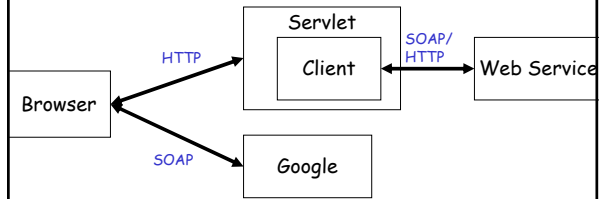
- Option 2: Web application



37

Web Application Architecture

- Web Service
 - Option 3: Web application + AJAX



38