CS 683 Emerging Technologies
Fall Semester, 2006
Doc 25 Mashups, XMLRPC & SOAP
Nov 28, 2006

References

http://www.xmlrpc.com/ Main XML_RPC web site

Wikipedia, Mashup, XMLPRC, SOAP, REST, http://en.wikipedia.org/wiki/Main_Page

Amazon REST Services, http://docs.amazonwebservices.com/AWSEcommerceService/2006-11-14/PgRestRequestsArticle.html

Amazon SOAP Services, http://docs.amazonwebservices.com/AWSEcommerceService/2006-11-14/PgSoapRequestsArticle.html

Mashup

Website “that seamlessly combines content
from more than one source into an integrated experience”

Wikipedia

Demos

http://www.housingmaps.com/

http://www.chicagocrime.org/map/

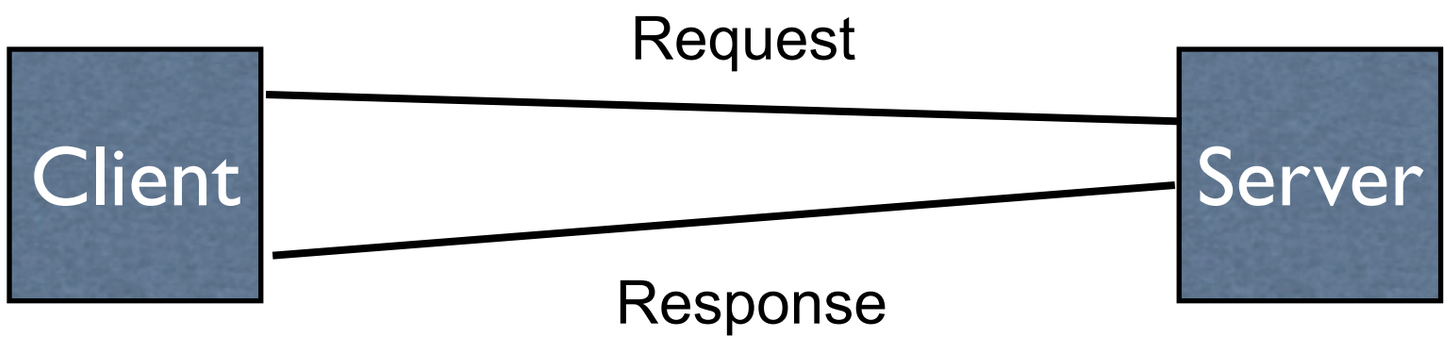
Systems for Remote Procedures

Remote Procedure Call

Allows a computer program running on one computer to
run a subroutine on another computer
without the programmer explicitly coding the details for this interaction

RFC 707	1976
Xerox Courier	1981
Sun ONC or RPC	1984
Microsoft Remote Procedure Call	
Microsoft .NET	2002

Basic RPC Interaction



Program does not deal with network code

CORBA

Common Object Request Broker Architecture

Allows programs to
call methods on objects on remote machines

Client and server code written in different languages

Handles network code for programmers

Supported languages
Ada, C, C++, Lisp, Smalltalk, Java, COBOL, PL/1, Python
Perl, Visual Basic, Tcl

1991 CORBA 1.0
1996 CORBA 2.0
2002 CORBA 3.0

Java's RMI

RMI in Java

RMI Server

```
public class HelloServer
        extends UnicastRemoteObject
        implements Hello
    {

    public HelloServer() throws RemoteException
        {
        }

    // The actual remote sayHello
    public String sayHello() throws RemoteException
        {
        return  "Hello World from " + getUnixHostName();
        }
```

XML

XML is about

- Document structure
- Describing data as strings

```
<?xml version="1.0" ?>
<CATALOG>
  <CD>
    <TITLE>Empire Burlesque</TITLE>
    <ARTIST>Bob Dylan</ARTIST>
    <COUNTRY>USA</COUNTRY>
    <COMPANY>Columbia</COMPANY>
    <PRICE>10.90</PRICE>
    <YEAR>1985</YEAR>
  </CD>
  <CD>
    <TITLE>Hide your heart</TITLE>
    <ARTIST>Bonnie Tyler</ARTIST>
    <COUNTRY>UK</COUNTRY>
    <COMPANY>CBS Records</COMPANY>
    <PRICE>9.90</PRICE>
    <YEAR>1988</YEAR>
  </CD>
</CATALOG>
```

XML-RPC

RPC using
HTTP as transport layer and
XML to encode request/response
Language and platform independent

Started by Userland (http://frontier.userland.com/) in 1998

Example - Adding Two Numbers

Client

```
import java.util.*;
import org.apache.xmlrpc.*;

public class XmlRpcExample {
        public static void main (String args[]) {
        try {
            XmlRpcClient xmlrpc =  new XmlRpcClientLite
                                    ("http://xmlrpc.usefulinc.com/demo/server.php");
                    Vector parameters = new Vector ();
            parameters.addElement (new Integer(5) );
            parameters.addElement (new Integer(3) );
            Integer sum =
                                (Integer) xmlrpc.execute ("examples.addtwo", parameters);
            System.out.println( sum.intValue() );  }
        catch (Exception serverProblem) {
            serverProblem.printStackTrace( System.out);
        }
    }
}
```

Add-Two Server

```
import org.apache.xmlrpc.*;

public class JavaServer  {
        public Integer addtwo(int x, int y)   {
        return new Integer( x + y);
    }

    public static void main( String[] args)  {
        try {
            System.out.println("Starting server on port 8080");
            WebServer addTwoServer = new WebServer(8080);
            addTwoServer.addHandler("examples", new JavaServer());
            System.out.println("server running"); }
        catch (Exception webServerStartError) {
            System.err.println( "JavaServer " + webServerStartError.toString());
        }
    }
}
```

Message sent to Server

```
POST /demo/server.php HTTP/1.1
Host: xmlrpc.usefulinc.com
Content-length: 190
Content-type: text/xml;charset=iso-8859-1
User-Agent: Smalltalk XMLRPC version 0.5 (VisualWorksÆ NonCommercial, Release 7 of
June 14, 2002)
Connection: keep-alive

<?xml version="1.0"?>
<methodCall>
	<methodName>examples.addtwo</methodName>
	<params>
		<param>
			<value><int>5</int></value>
		</param>
		<param>
			<value><int>3</int></value>
		</param>
	</params>
</methodCall>
```

How Does this Work?

Client marshals (serialize) the rpc request

Converts the requests in to a format that can be sent on the network

Client

Sends the marshaled version to the server
Waits for server response

Server

Unmarshals the request,
Runs the requested method
Marshals the result
Send the marshaled result back to the client

Client unmarshals the result

XML-RPC Supported Data-Types

<i4> or <int>
<boolean>
<string>
<double>
<dateTime.iso8601>
<struct>
<array>
<base64>

Web Services

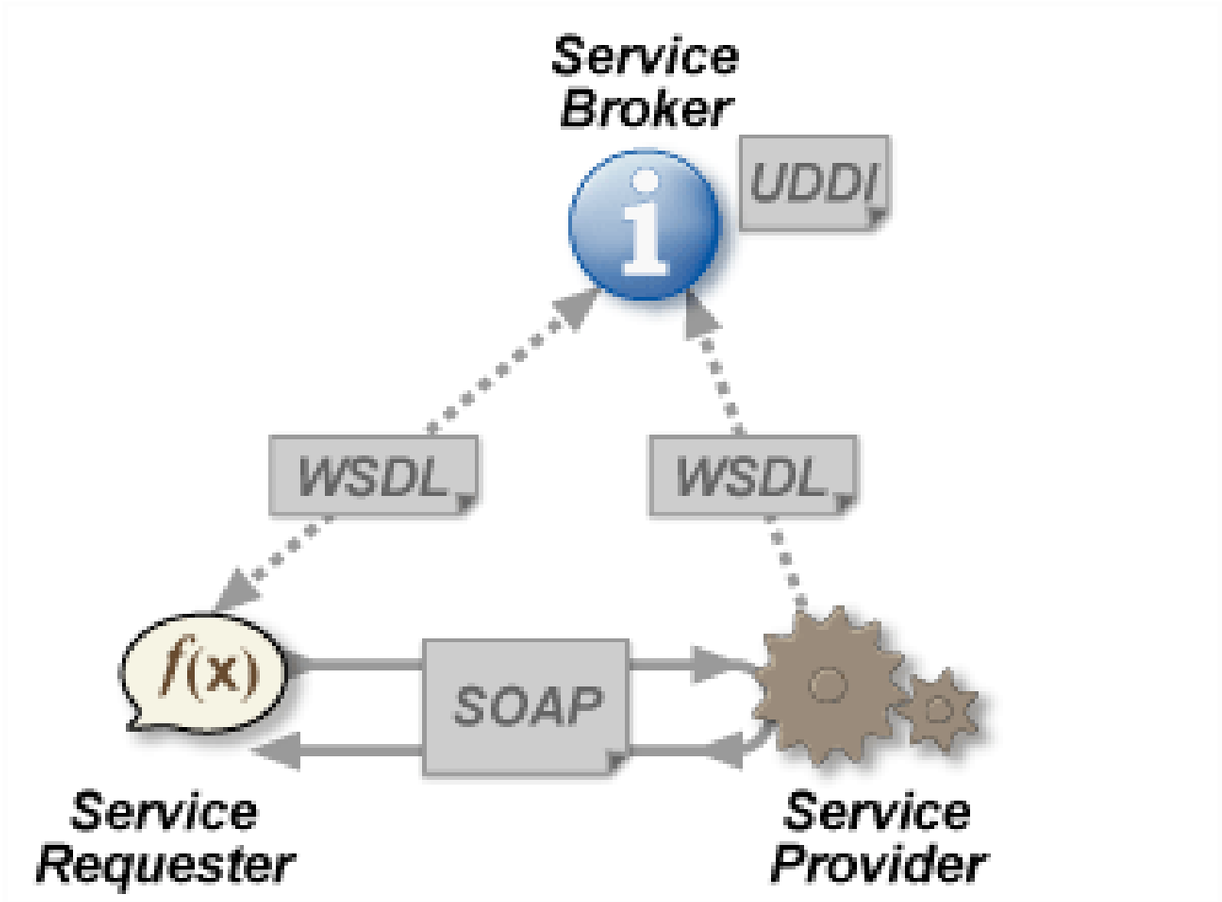


Image from
http://en.wikipedia.org/wiki/Web_service

Terms

SOAP - Simple Object Access Protocol (1998)

- Remote Procedure call
- Transport - HTTP & SMTP
- Request encoded using XML
- Can send objects
- Provides a service

WSDL - Web Services Description Language

- XML description of SOAP calls
- Procedure name & arguments
- Return type
- Location of service

UDDI - Universal Description, Discovery, & Integration

- Information about services

Simple Add Example in Ruby

AddTwoCalculator.rb

```
class AddTwoCalculator
  attr_reader :call_count

  def initialize
    @call_count = 0
  end

  def add(a, b)
    @call_count += 1
    return a + b
  end

end
```

AddTwo Server

AddTwoServer.rb

```
require 'soap/rpc/standaloneServer'
require 'addtwocalculator'

Addtwo_Name_Space = 'http://www.eli.sdsu.edu/addtwo'

class AddTwoServer < SOAP::RPC::StandaloneServer
  def on_init
    adder = AddTwoCalculator.new
    add_method(adder, 'add', 'first', 'second')
    add_method(adder, 'call_count')
  end
end

server = AddTwoServer.new('Adder', Addtwo_Name_Space, '0.0.0.0', 8765)
trap('INT') {server.shutdown}
server.start
```

AddTwo Client

AddTwoClient.rb

```
require 'soap/rpc/driver'

endpoint = 'http://127.0.0.1:8765'
namespace = 'http://www.eli.sdsu.edu/addtwo'

addProxy = SOAP::RPC::Driver.new(endpoint, namespace)
addProxy.add_method('add', 'first', 'second')
addProxy.add_method('call_count')

puts "Call count: #{addProxy.call_count}"
puts addProxy.add(1, 2)
puts "Call count: #{addProxy.call_count}"
```

Web Service Example - BabelFish

Provide

- Source Language
- Destination Language
- Text

BabelFish converts text

Basic Steps

Request WSDL of service

Use WSDL to formulate request

Send request over http

Receive result

Smalltalk Client

```
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.
```

BabelFish WSDL

```
<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/"
        xmlns="http://schemas.xmlsoap.org/wsdl/">

        <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>
</definitions>
```

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>
```

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>
```

Performance

	Connect Time	Send String 21,000 chars	Send String Relative time	Send 5,000 integers	Relative Performance
Raw Sockets	0.002242	0.001377	1	6.7	4.5
CORBA	0.000734	0.004601	3.4	1.5	1
XML-RPC	0.007040	0.082755	60.1	100.3	66.7
SOAP	0.000610	0.294198	213.7	1324.3	882.9

Python implementations comparied

http://www-128.ibm.com/developerworks/webservices/library/ws-pyth9/

REST

Representational State Transfer

Roy Fielding, 2000 UCI Ph.D. Thesis

Basic Structure

URL points to a resource

RPC vs REST

RPC

```
class Server
    def getUser
    def addUser
    def findUser
    etc.
```

Sample client code

```
proxy = UserServer.new(http://foo.bar)
proxy.findUser('whitney')
```

REST

URL for each resource

http://foo.bar/users/

http://foo.bar/users/{user} (one for each user)

http://foo.bar/findUserForm

Sample client code

```
user = Browser.url('http://foo.bar/users/whitney')
user.get()
```

Pseudo-REST (POX/HTTP)

HTTP url returns plain old XML document

Program uses data in XML document

Amazon POX/HTTP - E-Commerce

Request

http://webservices.amazon.com/onca/xml?Service=AWSECommerceService
&AWSAccessKeyId=[Your Access Key ID Here]
&AssociateTag=[Your Associate ID Here]
&Operation=SimilarityLookup
&ItemId=0060006781
&ResponseGroup=Small

http://docs.amazonwebservices.com/AWSEcommerceService/2006-11-14/PgRestRequestsArticle.html

Amazon Response


```
<?xml version="1.0" encoding="UTF-8"?>
<SimilarityLookupResponse xmlns="http://webservices.amazon.com/AWSECommerceService/2006-10-31">
  <OperationRequest>
    <HTTPHeaders>
      <Header Name="UserAgent" Value="Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"/>
    </HTTPHeaders>
    <RequestId>1CJZPMVXXRNS84XHHJQR</RequestId>
    <Arguments>
      <Argument Name="Service" Value="AWSECommerceService"/>
      <Argument Name="AssociateTag" Value="[Your Associate ID Here]"/>
      <Argument Name="AWSAccessKeyId" Value="[Your Access Key ID Here]"/>
      <Argument Name="ItemId" Value="0060006781"/>
      <Argument Name="ResponseGroup" Value="Small"/>
      <Argument Name="Operation" Value="SimilarityLookup"/>
    </Arguments>
  </OperationRequest>
  <Items>
    <Request>
      <IsValid>True</IsValid>
    </Request>
```

Amazon Response - Continued


```
<Item>
  <ASIN>0060932902</ASIN>
  <DetailPageURL>http://www.amazon.com/gp/redirect.html?location=/exec/obidos/ASIN/0060932902/[Your Associate ID Here]%3FAWSAccessKeyId=[Your Access Key ID Here]%26camp=2025%26link_code=xm2</DetailPageURL>
  <ItemAttributes>
    <Author>Matt Ridley</Author>
    <ProductGroup>Book</ProductGroup>
    <Title>Genome</Title>
  </ItemAttributes>
</Item>
<Item>
  <ASIN>0140264450</ASIN>
  <DetailPageURL>http://www.amazon.com/gp/redirect.html?location=/exec/obidos/ASIN/0140264450/[Your Associate ID Here]%3FAWSAccessKeyId=[Your Access Key ID Here]%26camp=2025%26link_code=xm2</DetailPageURL>
  <ItemAttributes>
    <Author>Matt Ridley</Author>
    <ProductGroup>Book</ProductGroup>
    <Title>The Origins of Virtue: Human Instincts and the Evolution of Cooperation</Title>
  </ItemAttributes>
</Item>
   etc.
```

Amazon SOAP - E-Commerce


```
<?xml version="1.0" encoding="UTF-8"?>
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://webservices.amazon.com/AWSECommerceService/2006-11-14" targetNamespace="http://webservices.amazon.com/AWSECommerceService/2006-11-14">
  <types>
        <xs:schema targetNamespace="http://webservices.amazon.com/AWSECommerceService/2006-11-14" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://webservices.amazon.com/AWSECommerceService/2006-11-14" elementFormDefault="qualified">

        <xs:element name="Bin">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="BinName" type="xs:string" />
                    <xs:element name="BinItemCount" type="xs:positiveInteger" />
                    <xs:element name="BinParameter" minOccurs="0" maxOccurs="unbounded">
                        <xs:complexType>
                          <xs:sequence>
                              <xs:element name="Name" type="xs:string" />
                              <xs:element name="Value" type="xs:string" />
                          </xs:sequence>
                        </xs:complexType>
                    </xs:element>
```


+ 3,000 more lines