**CS 580 Client-Server Programming**
**Spring Semester, 2004**
**Doc 2 XML-RPC**

RPC ..... 2
XML-RPC ..... 3
Java Client ..... 6
Smalltalk Client ..... 8
Issues ..... 9
XmlRpc Servers ..... 13
Java Example ..... 13
Smalltalk Server Example ..... 14

# References

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

http://davenet.userland.com/1998/07/14/xmlRpcForNewbies XML_RPC For Newbies

http://xmlrpc-c.sourceforge.net/xmlrpc-howto/xmlrpc-howto.html XML-RPC How to

http://xml.apache.org/xmlrpc/ Home page for Java XML-RPC implementation

Some XML-RPC Services
http://www.stuffeddog.com/speller/doc/rpc.html Spell Checker

http://www.xmlrpc.com/currentTime Current time

http://www.dscpl.com.au/xmlrpc-debugger.php List of several services

**Copyright** ©, All rights reserved. 2004 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.

# RPC
# Remote Procedure Call

A client can "directly" call a function or procedure on the server

## Issues

- Cross platform
  Primitive data types may be different on client & server

- Marshalling/unmarshalling of parameters and results
  Procedure parameters must be sent from client to server
  How can one handle pointers as parameters?
  Result of procedure call must be sent back to client

- Different contexts of client and server

- Registering and finding servers

## Sample Uses

Unix NFS (Network File System)
Unix license managers

## RPC implementations

SUN RPC
Distributed Computing Environment (DCE)

# 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

Languages/Systems with XML-RPC implementations

- Java, Perl, Python, Tcl, C, C++, Smalltalk
- ASP, PHP, AppleScript, COM
- Zope, WebCrossing

Led to the development of SOAP

# Smalltalk Example

```
| client sum |
client :=XmlRpcClient url: 'http://xmlrpc.usefulinc.com/demo/server.php'.
sum := client
        perform: 'examples.addtwo'
        with: 5
        with: 3.
^sum
```

# Java Example

```
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 (java.net.MalformedURLException badAddress)
         {
         badAddress.printStackTrace( System.out);
         }
      catch (java.io.IOException connectionProblem)
         {
         connectionProblem.printStackTrace( System.out);
         }
      catch (Exception serverProblem)
         {
         serverProblem.printStackTrace( System.out);
         }
      }
   }
```

# Some Client Details
# Java Client

Main Page: http://www.xmlrpc.com/
Download page: http://xml.apache.org/xmlrpc/download.html
JavaDoc: http://xml.apache.org/xmlrpc/apidocs/index.html

## Two Clients

XmlRpcClient
- Uses java.net.URLConnection
- Supports proxies, redirects, cookies

XmlRpcClientLite
- Contains its own http client implementation
- Does not support proxies, redirects, cookies
- Faster than XmlRpcClient
- Supports same methods as XmlRpcClient

# Important Methods

XmlRpcClient(String serverURL)

Returns a client which interacts with the indicated server

Object execute(String method, Vector parameters)

Executes the method with the parameters on the server
Server must be written to support the method
Returns the result

Throws

IOException if can not connect to server from some reason

XmlRpcException if server has problem executing method

# Smalltalk Client

Download page:
http://www.eli.sdsu.edu/SmalltalkCode/xmlrpc/index.html

XmlRpcClient class>>url: serverURL

Returns a client which interacts with the indicated server

XmlRpcClient>>perform: aMethodName

XmlRpcClient>>perform: aMethodName with: oneArgument

XmlRpcClient>>
perform: aMethodName
with: firstArgument
with: secondArgument

XmlRpcClient>>
perform: aMethodName withArguments: collectionOfArguments

Executes the method with the parameters on the server
Server must be written to support the method
Returns the result

The perform: methods throw

XmlRpcParseError
Error

# Issues

Client program has to know

- Server machine name or IP (xmlrpc.usefulinc.com)
- Path to server program (/demo/server.php)
- Name of remote method (examples.addtwo)
- Number, Type and Order of arguments

## Supported Data Types – Client Side

| **XML-RPC data type** | **Java** | **Smalltalk** |
|---|---|---|
| <i4> or <int> | java.lang.Integer | SmalltInteger |
| <boolean> | java.lang.Boolean | true, false |
| <string> | java.lang.String | String |
| <double> | java.lang.Double | Float |
| <dateTime.iso8601> | java.util.Date | Timestamp |
| <struct> | java.util.Hashtable | Dictionary |
| <array> | java.util.Vector | OrderedCollection |
| <base64> | byte[ ] | ByteArray |

## Additional Smalltalk To XML-RPC Mappings

| Smalltalk | XML-RPC |
|---|---|
| Integer (including Large) | <int> |
| Number | <double> |
| SequenceableCollection | <array> |

## Extended XML-RPC Types in Smalltalk implementation

| Smalltalk | XML-RPC |
|---|---|
| Object | <object> |
| Nil | <nil> |

Support for the Object type is limited to objects with no circular references.

## How do you know about methods on the Server?

The server:

- url
- method name
- method arguments

need to be documented some place

# 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

# Complete Request 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>
```

# XmlRpc Servers
# Java Example

The following starts an addtwo server on port 8080
Server URL is serverMachinename:8080
Method name is: examples.addtwo

How come the server is still running after the last println?

```
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());
         }
      }
   }
```

# Smalltalk Server Example

```
Smalltalk defineClass: #WaveXmlRpcAddtwo
   superclass: #{Smalltalk.WaveXmlRpcServer}
   indexedType: #none
   private: false
   instanceVariableNames: ''
   classInstanceVariableNames: ''
   imports: ''
   category: 'Network-XMLRPC-Server'


WaveXmlRpcAddtwo class methodsFor: 'utility'

allowedServerMethods
   "Answer a dictionary
   Key is the string the client will use to request the method
   Value is the symbol for the method to actually call"

   ^(Dictionary new)
      at: 'examples.addtwo' put: #add:to:;
      yourself


WaveXmlRpcAddtwo methodsFor: 'accessing'

add: anInteger to: aSecondInteger
   ^anInteger + aSecondInteger
```

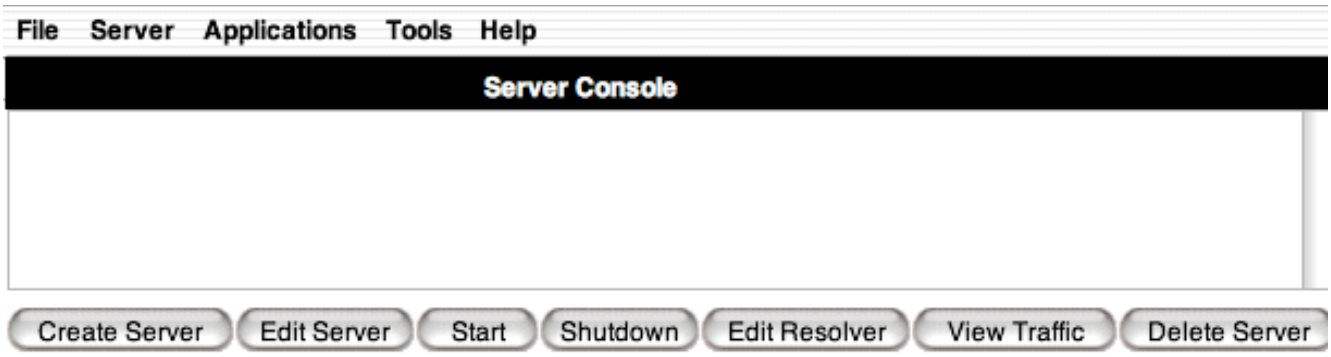
# Notice

We have not explicitly handled sockets in any example

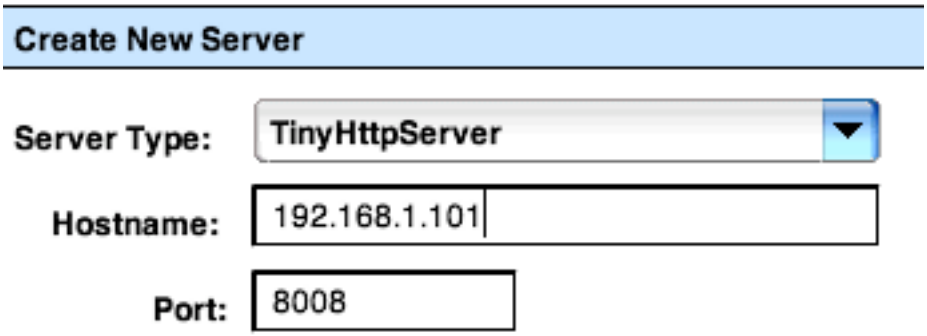
# How to Start the Smalltalk Server

File in the XmlRpcServer parcel

The parcel contains the WaveXmlRpcAddtwo class and loads the Wave web server. When this is done the Server Console window will open up. It looks like:

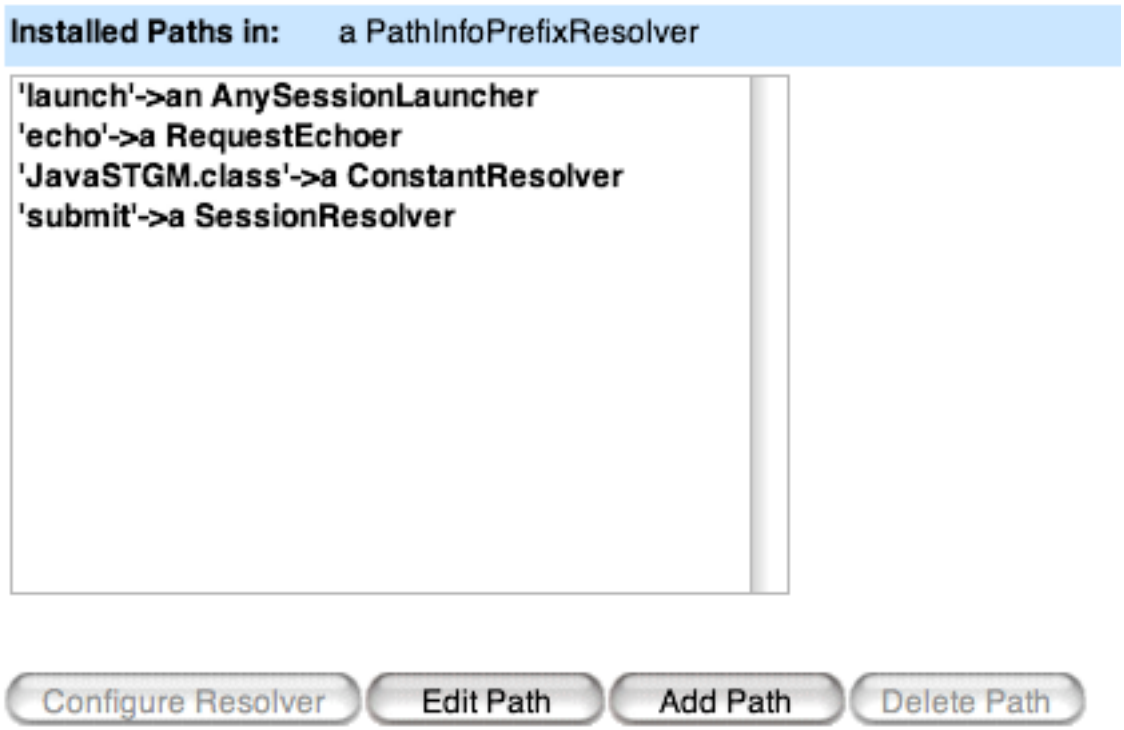



Click on the “Create Server” button. The window will expand as seen below.

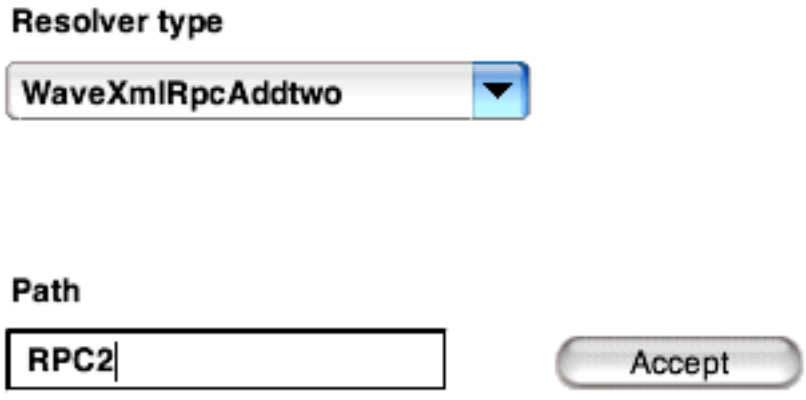



Set the server type to “TinyHttpServer”, set the Hostname to the name or IP of the machine you are running the server on, set the port to the port you wish the server to listen to. Then click the “Create and Start” button on the bottom of the window.

Now click on the “Edit Resolver” button. The bottom of the window will expand to look like:




Click on the “Add Path” button.




Set the “Resolver type” to WaveXmlRpcAddtwo. Note any subclass of WaveXmlRpcServer will appear in the resolver type drop down menu. Add a path. I used RPC2, which was common when XML-RCP was new. Click on the “Accept” button.

In the above example
Server URL is: http://192.168.1.101:8008/RPC2
Method name is: examples.addtwo

This sever is not running so you will not be able to connect to it.