

XML-RPC and SOAP

XML messaging

- RPC: XML is used to encode procedure calls and responses.
- EDI: Electronic Document Interchange
 - transfer documents between applications across a network
 - purchase orders, financial transactions, etc.

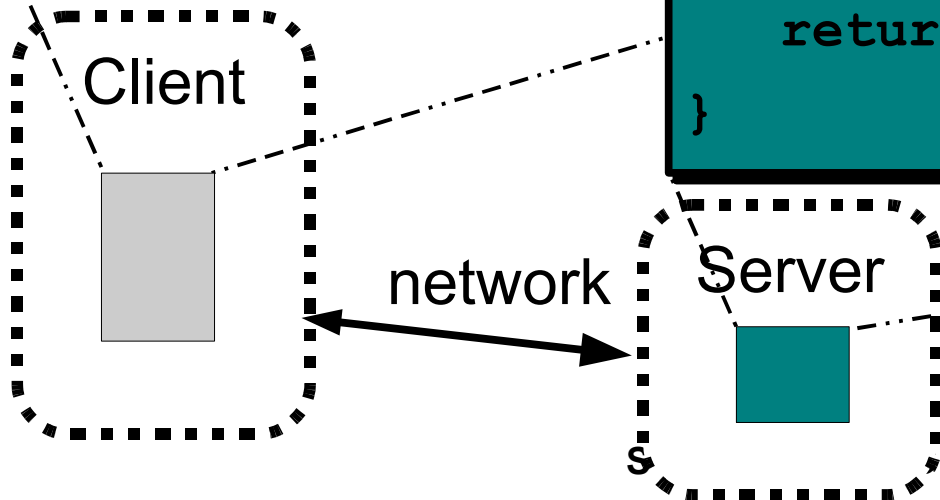
RPC: Remote Procedure Call

- Model interprocess communication as procedure calls.
 - request: pass parameters
 - response: return value(s)
- There have been a variety of RPC implementations supporting varying languages: Sun RPC (C), RMI (JAVA), Windows RPC, ...

RPC (general idea)

```
get_user_record(uid) {  
    request.param = uid;  
    request.function = "lookupUser";  
    response = call_rpc("foo.com", request);  
    return response.user_rec;  
}
```

```
lookupUser(uid) {  
    query = "SELECT * FROM ...."  
    ...  
    return(record);  
}
```



XML-RPC

- Use XML to *encode* requests
 - procedure name
 - parameter values
- Response is also an XML document
 - return value(s)
 - errors (faults)
- Both are well defined document types
 - tag names are defined in the XML-RPC specification document.

Uses HTTP POST

- Use existing protocol (and software!).
- Avoid firewall issues (everyone allows HTTP traffic).
- XML-RPC Request is the body of an HTTP POST.
- XML-RPC Response is the body (content) of the HTTP response.

Example Request

(swiped from xml-rpc.com)

POST /RPC2 HTTP/1.0

Host: betty.userland.com

User-Agent: Frontier/5.1.2 (WinNT)

Content-Type: text/xml

Content-length: 181

```
<?xml version="1.0"?>
```

```
<methodCall>
```

```
  <methodName>examples.getStateName</methodName>
```

```
  <params>
```

```
    <param>
```

```
      <value><i4>41</i4></value>
```

```
    </param>
```

```
  </params>
```

```
</methodCall>
```

Sample Response

```
HTTP/1.1 200 OK
Connection: close
Content-Length: 158
Content-Type: text/xml
Date: Fri, 17 Jul 1998 19:55:08 GMT
Server: UserLand Frontier/5.1.2-WinNT<p>
```

```
<xml version="1.0"?>
<methodResponse>
  <params>
    <param>
      <value>
        <string>South Dakota</string>
      </value>
    </param>
  </params>
</methodResponse>
```

XML-RPC Data Types

`<int>` *or* `<i4>`

`<boolean>`

`<string>`

`<double>`

`<dateTime.iso8601>`

`<struct>`

`<array>`

XML-RPC struct

```
<struct>
  <member>
    <name>Hostname</name>
    <value>
      <string>monte.cs.rpi.edu</string>
    </value>
  </member>
  <member>
    <name>IPAddress</name>
    <value>
      <string>128.213.7.32</string>
    </value>
  </member>
</struct>
```

XML-RPC array

```
<array>  
  <data>  
    <value><i4>12</i4></value>  
    <value><string>Egypt</string></value>  
    <value><boolean>0</boolean></value>  
    <value><i4>-31</i4></value>  
  </data>  
</array>
```

XML-RPC Programming

- Need to be able to generate HTTP requests (client) and responses(server).
- Need to generate XML documents.
- Need to parse XML documents and extract specific items.
- Need to handle faults (errors).

XML-RPC Libraries

- There are many libraries that provide most of what we need:
 - all the XML stuff.
 - all the networking (HTTP).
- By developing services based on XML-RPC we can extract information from a remote server with much less code (basically avoid the network and XML coding).

Practical Issues

- There are many public XML-RPC based web services, but in general we can use it to develop our own web services
 - our web applications are *clients*
 - we also write the xml-rpc servers.

SOAP

Simple Object Access Protocol

- Same general idea as XML-RPC, but more features:
 - enumerations
 - Polymorphism (type determined at run time)
 - user defined data types

SOAP

- XML Documents are more complex
 - use namespaces
 - formal "envelope"
 - Soap Header
 - Soap Body

SOAP Request Example

```
POST /StockQuote HTTP/1.1
Host: www.stockquoteserver.com
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAPAction: "Some-URI"
```

```
<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>
```

XML-RPC vs. SOAP

- XML-RPC is much simpler
- There are lots of *web services* based on XML-RPC.
- SOAP makes it easier to exchange more complex documents.
- SOAP runs over many protocols:
 - HTTP, SMTP, FTP, ...

SOAP is an integral part of .NET

Programmer Transparency
Multiple Languages

PHP and XML-RPC

- There are a number of options:
 - write it yourself (HTTP and XML stuff).
 - php supports an *experimental* implementation as an extension (which you probably don't have!).
 - php libraries:
 - XML-RPC for PHP by Edd Dumbill

XML-RPC for PHP Library

Available via

sourceforge.net/projects/phpxmlrpc

or here:

[xmlrpc-1.2.1.zip](#)

unzip in a new folder, and copy the files

xmlrpc.inc and **xmlrpcs.inc** into your
php folder.

XML-RPC Sample

- Server provides a *remote addition* procedure:
 - expects two integers as parameters
 - returns the sum.
- Wow – what a novel idea!
- Client (PHP program) allows user to type numbers into a form.

Relevant Client Code

(see the complete code for details!)

```
$params = array(new xmlrpcval($x, "double"),
                new xmlrpcval($y, "double"));
$msg = new xmlrpcmsg('remoteAdd', $params);

$client = new xmlrpc_client("/addserv.php",
                           "localhost", 80);

$response = $client->send($msg);
... check for errors ...
$val = $response->value();
$sum = $val->scalarval();
```

Server Code (abbreviated)

```
function sum($params) {  
    $xv = $params->getParam(0);  
    $x = $xv->scalarval();  
    $yv = $params->getParam(1);  
    $y = $yv->scalarval();  
    $z = $x + $y;  
    $retval = new xmlrpcval($z, "double");  
    return new xmlrpcresp($retval);  
}  
$method_map = array("remoteAdd" =>  
                    array("function" => "sum") );  
$s = new xmlrpc_server($method_map);
```

Complete Examples

- The add client/server:
 - [addclient.php](#)
 - (needs xmlrpc.inc)
 - [addserv.php](#)
 - (needs xmlrpc.inc and xmlrpcs.inc)
- Check out the examples in the xmlrpc for php library as well... (in the zip file).

Exercise

- Write an XML-RPC client that uses my XML-RPC server at:

<http://cgi2.cs.rpi.edu/~hollingd/eiwfortune/>

- The server has a procedure named `getWittyQuote` that expects a single parameter of type `int`:

0 : returns a random Star Trek quote.

1 : returns a random "Real Programmers..." quote.

2 : returns a random Murphy's law variant.

(all taken from the Unix fortune database).

Debugging

You can view your XML request before sending it:

```
$msg = new xmlrpcmsg('getWittyQuote', $params);  
  
print "<pre>";  
print htmlentities($msg>serialize());  
print "</pre>\n";
```

You can turn on debugging:

```
$client = new xmlrpc_client( ... );  
$client->setDebug(1);
```

Resources

- xml-rpc.com (userland.com)
 - specification, info, tutorials, code resources
seems to have gone away...
- www.soapware.org
 - tutorials, code resources
- www.w3.org
 - SOAP specification