XML-RPC

Requesting Remote Services

Adrian Dydecki
Version: 1.0 November 2009

TABLE OF CONTENTS:

1 Overview

"XML-RPC is a lightweight Internet protocol for requesting remote services. All requests and responses are expressed as XML documents carried by HTTP messages" [2]. XML-RPC is a remote procedure call using HTTP (Hypertext Transfer Protocol) as the transport layer and XML (eXtensible Markup Language) as the encoding. An XML-RPC call is always between two parties: the client (requesting process) and the server (responding process). Figure 1 shows basic concept of remote procedure call using XML-RPC protocol.

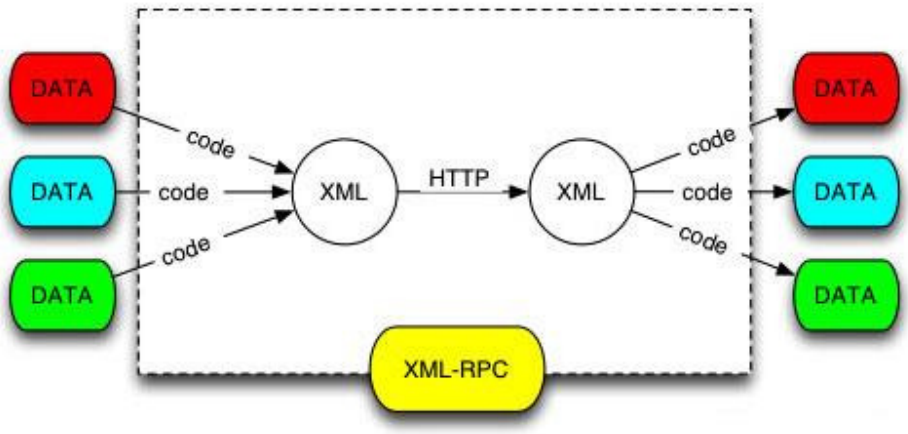

Figure 1 Remote procedure call
Source: [http://www.xmlrpc.com]

Server is available at a particular URL (Uniform Resource Locator) e.g. http://xmlrpcserver:8080/ which means that this is HTTP server and responding on port 8080. To use remote procedure available on the server, the following steps are necessary:

1. The client program makes a procedure call using XML-RPC client module, specifying a procedure name, parameters, and a target server.
2. The XML-RPC client module takes the procedure name and parameters and then packages them as XML. Then the client module issues an HTTP POST request containing the request information to the target server.
3. The target server receives the POST request and passes the XML content to an XML-RPC server module.

4. The XML-RPC server module parses the XML to get the procedure name and parameters and then calls the appropriate method on the server, passing it the parameters.
5. The method returns a response to the XML-RPC server module and the XML-RPC server module packages the response as XML.
6. The server returns that XML as the response to the HTTP POST request.
7. The XML-RPC client module parses the XML to extract the return value and then passes the return value back to the client program.
8. The client program continues processing with the return value.

These steps are more explained in the book Laurent S.St., "Programming Web Services with XML-RPC", O'Reilly, 2001. Figure 2 shows these steps.

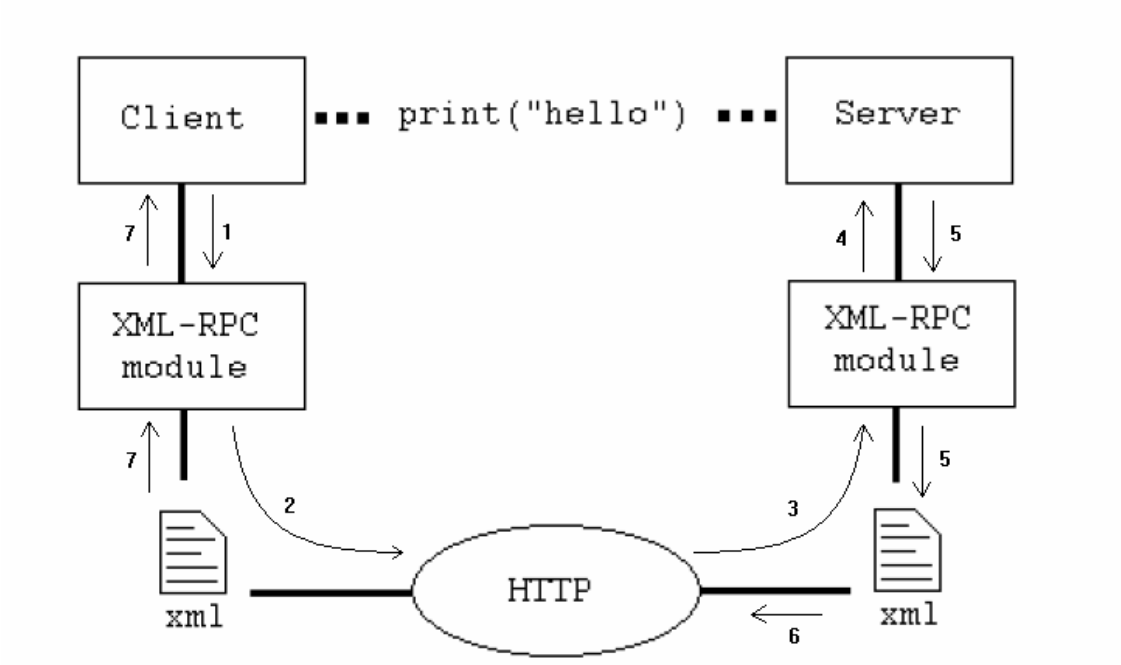

Figure 2 Calling print ("hello") on the server using XML-RPC protocol

Using HTTP as the transport layer means that XML-RPC calls must be synchronous and stateless.

1.1 Synchronous calls

The synchronous calls mean that client waits until it receives a response from the server. This happens because the response must occur on the same HTTP connection as

the request. This has consequences for program design – server should execute XML-RPC method in reasonable time, and the client should be ready for potential blocking while invoking XML-RPC procedure.

1.2 Stateless calls

If the client program invokes one procedure on the server, and invokes it again, XML-RPC server treats them as two isolated procedures not connected each other.

2 Data types

Procedure parameters can be simple data types like numbers, strings, dates, etc.; and can also be complex record and list structures [1]. To represent these values an XML-RPC protocol defines an XML representation for the program data. Any data item in an XML-RPC request and response is contained within a <value> ... </value> element [3, pages 18-23]. Table below describes simple data types.

Tag	Type	Example
<i4> or <int>	32-bit signed integer	1234
<double>	double precision signed floating point number	-12.34
<boolean>	boolean logical value: true (1) or false (0)	1
<string>	string	Hello, World!
<dateTime.iso8601>	date/time	20080311T15:17:35
<base64>	base64-encoded binary	ksdSEK23789dsKJsafEn==

Table 1 Simple data types

2.1 Integers

An XML-RPC integer is represented as 32-bit signed integer. It has two representations:

```
<value><i4>n</i4></value>
```

or:

```
<value><int>n</int></value>
```

XML-RPC implementation must recognize both of these representations. The string representing integer value can contains only characters 0 to 9, '-'and '+'. A positive integer may optionally have the plus sign.

2.2 Floating-point numbers

Float numbers are representing within <double> ... </double> element. XML-RPC uses 64 bits for the representations of float numbers. This means that range of the floating-point numbers is as double precision floating point. This gives us range of ± ~$10^{-323.3}$ to ~$10^{303.3}$. An example below shows how to use this data type.

```
<value><double>n</double></value>
<value><double>123.321</double></value>
<value><double>-0.987</double></value>
```

2.3 Boolean values

Boolean data has one of two values, true or false. In XML-RPC the value 1 corresponds to the Boolean value true, and 0 to the Boolean value false. Below are all the possible representations.

```
<value><boolean>1</boolean></value>
```

or:

```
<value><boolean>0</boolean></value>
```

2.4 Strings

A string has two representations in XML-RPC protocol. The first one is enclosed within the <value>...</value> element, it means that everything between <value>...</value> element is string. Below is the example.

```
<value>Hello, World!</value>
```

The second one is enclosed within <string>...</string> element. Below is the example.

```
<value><string>Hello, World!</string></value>
```

The XML-RPC protocol must understand both representations. If you wish to use special characters, e.g. <, >, &, you must use predefined entity reference, e.g. & for &.

2.5 Date-times

Dates and times are encoded in ISO 8601 standard. Dates and times are enclosed within <dateTime.iso8601>...</dateTime.iso8601> element. A date-time in XML-RPC is represented as follows:

```
<dateTime.iso8601>CCYYMMDDTHH:MM:SS</dateTime.iso8601>
<dateTime.iso8601>20080311T15:17:35</dateTime.iso8601>
```

2.6 Binary

The <base64> ... </base64> element is used to enclose the binary objects. Binary data type is used when we want to transfer data items such as binary files, e.g. images. The encoding is defines in RPC 2045, available at http://ietf.org/rfc/rfc2045.txt. More information about base64 encoding can be found at http://en.wikipedia.org/wiki/Base64.

2.7 Arrays

Array data types are representing in the following format:

```
<value>
 <array>
  <data>
   <value> ... </value>
   ...
   <value> ... </value>
  </data>
 </array>
</value>
```

All items within array data type can be simple XML-RPC data types or complex record and list structures. It is possible to represent multidimensional arrays by

embedding array within an array. XML-RPC arrays are as untyped arrays, because their items can be different types.

2.8 Structs

Basically struct is used to declare new types. It is grouping variables together. In XML-RPC struct is represented as series of members. Each member is a pair: name and value. The name must be an ASCII string and a value may be any XML-RPC value. Struct data types are representing in the following format:

```
<value>
 <struct>
  <member>
   <name>NameA</name>
   <value>ValueA</value>
  </member>
  ...
  <member>
   <name>NameN</name>
   <value>ValueN</value>
  </member>
 </struct>
</value>
```

Struct data types with duplicate member names are illegal struct in XML-RPC protocol, but many XML-RPC implementations accept an illegal structs.

3 Request format

The XML-RPC request has two parts: HTTP headers and XML body. HTTP headers identify server, and the XML body contains information about what method to invoke.

3.1 HTTP Headers

HTTP headers identify the server. XML-RPC protocol requires User-Agent, Content-Length, Content-Type and Host headers. User-Agent is the name of the XML-RPC implementation. Content-Length is the length of the XML Body described in the next section. Content-Type is constant in XML-RPC protocol and is set to text/xml value. Host header specifies the name of the server that should service XML-RPC request. XML-RPC protocol uses HTTP POST method indicating how to send the data to the server. The following example shows HTTP headers in XML-RPC request:

```
POST HTTP/1.0
User-Agent: MyXMLRPC implementation
Host: http://localhost:8080/
Content-Type: text/xml
Content-Length: 314
```

3.2 XML Body

XML-RPC request is representing within <methodCall>...</methodCall> tag. The name of the XML-RPC procedure to invoke is enclosed than in <methodName>...</methodName> element. When invoking XML-RPC procedure it is possible to include parameters. All parameters are enclosed within <params>...<params> tag. Each parameter is an XML-RPC value defined in the previous chapter and must be enclosed within <param>...</param> element. The following example shows XML-RPC request format:

```
<?xml version="1.0"?>
<methodCall>
 <methodName>print</methodName>
 <params>
  <param>
   <value><string>Hello World!</string></value>
  </param>
 </params>
</methodCall>
```

This example above demonstrates function call print(“Hello World!”).

4 Response format

Like request, response is also packaged in two parts: HTTP headers and XML body. Unless there is no http error (like 404 Not Found or 403 Forbidden) or any low level errors, XML-RPC responses always uses the 200 OK HTTP status code even if a fault response is contained in the message.

4.1 HTTP Headers

Required fields are Content-Type and Content-Length. First one is always set to text/xml. The second depends on length of XML body. The following example shows HTTP headers in XML-RPC response:

```
HTTP/1.0 200 OK
Connection: close
Content-Length: 158
Content-Type: text/xml
Date: Fri, 17 Jul 2009 19:55:08 GMT
Server: XmlRpc server
```

If the Connection header is specified with the value "close" in either the request or the response header fields, it indicates that the connection should not be considered 'persistent' after the current request/response is complete. More information about 'persistent' connections can be found at website http://www.w3.org/Protocols/rfc2616/rfc2616-sec8.html. When Connection tag is not specified the default for HTTP 1.0 is to close the connection.

4.2 XML Body

XML-RPC response is representing within <methodResponse>…</methodResponse> tag. There can be two types of XML-RPC response: single param response and fault response.

4.2.1 Single param response

This type of response come when the XML-RPC remote procedure was found on the server, executed correctly, and returned results. Parameter returned from server is enclosed within <params><param>...</param><params> tags. There can be only one parameter returned. Parameter can by any of XML-RPC values described in Data types chapter. The following example shows XML-RPC single param response format:

```
<?xml version="1.0"?>
<methodResponse>
  <params>
    <param>
      <value><string>Hello World!</string></value>
    </param>
  </params>
</methodResponse>
```

There is also possible to return more than one parameter by using array as returned value.

4.2.2 Fault response

This type of response comes when the XML-RPC remote procedure was not properly executed. This can happen e.g. when XML-RPC procedure was not found on the server or too many/too few parameters has been passed. Falut response like single param response has only one parameter and looks like below:

```
<?xml version="1.0"?>
<methodResponse>
  <fault>
    <value>
      <struct>
        <member>
          <name>faultCode</name>
          <value><int>-1</int></value>
          </member>
        <member>
          <name>faultString</name>
          <value><string>Method not found!</string></value>
          </member>
        </struct>
      </value>
    </fault>
  </methodResponse>
```

Internet

[1] **http://www.xmlrpc.com**
[2] **http://www.ontosys.com/xml-rpc**

Literature

[3] "Programming Web Services with XML-RPC", O'Reilly, 2001