XML-RPC

Contenu: Caractéristiques, Modèle de données, Requêtes, Réponses, Erreurs, Exemple TP1.

I.FALL, DGI/ESP/UCAD/SN
Janvier 2016

Caractéristiques

- XML-RPC est un protocole basé dur XML pour l'échange d'information entre ordinateur à travers le réseaux
- Peut être considéré comme l'une des approches de Services Web les plus simples permettant l'appel de procédures distantes
- Utilise le protocole HTTP pour passer de l'information d'un client à un serveur
- Utilise un petit vocabulaire XML pour décrire la nature des requêtes et des réponses

Caractéristiques

- Un client spécifie le nom d'une procédure et ses paramètres dans une requête XML, et le serveur retourne une faute ou une réponse dans une réponse XML
- Les paramètres sont une simple liste de types et de contenus
  - Les structures et les tableaux sont les typles les plus complexes disponibles
- Pas de notion d'objets et aucun mécanisme permettant d'inclure de l'information utilisant un autre vocabulaire
- Publié en 1998 par UserLand Software qui l'a initialement implémenté dans Frontier.

Caractéristiques

- Permettre de facilement intégrer des environnements de traitement différents n'utilisant pas des structures de données complexes
  - Permettre, par exemple, à Java de parler à du Perl, Python, ASP, etc.
- Permettre, au sein d'un même environnement, de connecter des programmes exploitant des modèles de données différents
- Offrir aux intégrateur une opportunité d'utiliser un vocabulaire standard pour l'échange d'information

Le modèle de données

- La spécification XML-RPC définit
  - 6 types de données de base
    - int ou i4, double, boolean, string, dateTime.iso8601, base64
  - et 2 types complexes
    - array, struct

Le modèle de données

- Les 6 types de base

Types	Valeurs	Exemples
int ou i4	entiers sur 32 bit entre -2,147,483,648 et 2,147,483,647.	<int>27</int> <i4>27</i4>
double	nombres en virgule flottante sur 64 bit	<double>27.31415</double> <double>-1.1465</double>
boolean	true (1) ou false (0)	<boolean>1</boolean> <boolean>0</boolean>
…	…	…

Le modèle de données

- Les 6 types de base

Types	Valeurs	Exemples
string	Texte ASCII (certaines implém supporte l'Unicode)	<string>Hello</string> <string>folks! @</string>
dateTime.iso8601	Dates au format ISO8601: CCYYMMDDTHH:MM:SS	<dateTime.iso8601> 20021125T02:20:04 </dateTime.iso8601> <dateTime.iso8601> 20020104T17:27:30 </dateTime.iso8601>
base64	Information binaire encodée en Base 64, comme défini dans la RFC 2045	<base64> SGVsbG8sIFdvcmxkIQ== </base64>

Le modèle de données

- Les types de base
  - Ils sont toujours contenus dans un élément <value>
  - Les string peuvent être contenues dans un élément <value> en ométant l'élément <string>

Le modèle de données

- Les tableaux
  - Informations séquentielles
  - Contenus dans un élément <array> qui contien un élément <data> qui à son tour contien un ensemble d'éléments <value> qui eux contiennent les éléments du tableau
  - Comme les autres éléments, un élément <array> est contenu dans un élément <value>

Le modèle de données

- Un tableau de 4 string

```
<value>
    <array>
        <data>
            <value><string>This </string></value>
            <value><string>is </string></value>
            <value><string>an </string></value>
            <value><string>array.</string></value>
        </data>
    </array>
</value>
```

Le modèle de données

- Un tableau de 3 entiers

```
<value>
    <array>
        <data>
            <value><int>7</int></value>
            <value><int>1247</int></value>
            <value><int>-91</int></value>
        </data>
    </array>
</value>
```

Le modèle de données

- Un tableau à valeurs de types mixtes

```
<value>
    <array>
        <data>
            <value><boolean>1</boolean></value>
            <value><string>ESP</string></value>
            <value><int>7</int></value>
            <value><double>1247.34578</double></value>
            <value><int>-91</int></value>
        </data>
    </array>
</value>
```

Le modèle de données

- Un tableau multidimensionnel

```
<value>
    <array>
        <data>
            <value>
                <array>
                    <data>
                        <value><int>1</int></value>
                        <value><int>2</int></value>
                    </data>
                </array>
            </value>
            <value>
                <array>
                    <data>
                        <value><int>1</int></value>
                        <value><int>2</int></value>
                    </data>
                </array>
            </value>

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

Le modèle de données

- Une structure

```
<value>
    <struct>
        <member>
            <name>givenName</name>
            <value><string>Mamadou</string></value>
        </member>
        <member>
            <name>familyName</name>
            <value><string>Camara</string></value>
        </member>
    </struct>
</value>
```

Le format de requête XML-RPC

- Une requête XML-RPC est une combinaison
  - de contenu XML
    - Nom de la méthode appelée
    - Paramètres
  - et d'entêtes HTTP
    - Pour la transmission de la requête sur le Web
- Chaque requête contient un document XML
  - Racine: <methodCall>
  - Nom de la méthode: <methodeName>
  - Chaque paramètre décrit dans un élément <param>, le tout dans un élément <params>

Le format de requête XML-RPC

- Exemple d'une requête d'appel d'une méthode circleArea prenant comme paramètre le rayon de type double, le contenu XML ressemble à

```
<?xml version="1.0"?>
<methodCall>
    <methodName>circleArea</methodName>
    <params>
        <param><value><double>2.41</double></value> </param>
    </params>
</methodCall>
```

Le format de requête XML-RPC

- Exemple d'une requête d'appel d'une méthode circleArea prenant comme paramètre le rayon de type double
- Le contenu XML ressemble à

```
<?xml version="1.0"?>
<methodCall>
    <methodName>circleArea</methodName>
    <params>
        <param><value><double>2.41</double></value> </param>
    </params>
</methodCall>
```

Le format de requête XML-RPC

- Les entêtes HTTP associées à ces requêtes reflètent le contenu ainsi que le client
- Un template de base

POST /target HTTP 1.x

User-Agent: Identifier

Host: host.making.request

Content-Type: text/xml

Content-Length: length of request in bytes

Le format de requête XML-RPC

- Exemple: si la méthode circleArea est disponible sur un serveur écoutant sur /xmlrpc
- L'entête de la requête serait …

```
POST /xmlrpc HTTP 1.0
User-Agent: myXMLRPCClient/1.0
Host: 192.168.124.2
Content-Type: text/xml
Content-Length: 169
```

Le format de requête XML-RPC

- … et la requête globale serait alors

```
POST /xmlrpc HTTP 1.0
User-Agent: myXMLRPCClient/1.0
Host: 192.168.124.2
Content-Type: text/xml
Content-Length: 169
<?xml version="1.0"?>
<methodCall>
      <methodName>circleArea</methodName>
      <params>
            <param><value><double>2.41</double></value> </param>
      </params>
</methodCall>
```

Le format de réponse XML-RPC

- Lorsque que la réponse est un succès …
  - Méthode trouvée, exécutée correctement, puis retourne un résultat
- … alors elle ressemble à une requête
  - Avec un remplacement de l'élément <methodCall> par un élément <methodResponse> et une omission de l'élément <methodName>

```
<?xml version="1.0"?>
<methodResponse>
    <params>
        <param>
            <value><double>18.24668429131</double></value>
        </param>
    </params>
</methodResponse>
```

Le format de réponse XML-RPC

- Une réponse contient un seul paramètre
  - Peut être un tableau ou une structure
  - Il est obligatoire de retourner une valeur dans une réponse
    - Même si c'est un boolean mis à 1 pour indiquer un succès côté serveur

Le format de réponse XML-RPC

- Une réponse contient des entêtes
  - Toutes les réponses XML-RPC utilisent le code de réponse 200 OK
- Entêtes types

```
HTTP/1.1 200 OK
Date: Sat, 06 Oct 2001 23:20:04 GMT
Server: Apache.1.3.12 (Unix)
Connection: close
Content-Type: text/xml
Content-Length: 124
```

Le format de réponse XML-RPC

- La réponse complète

```
HTTP/1.1 200 OK
Date: Sat, 06 Oct 2001 23:20:04 GMT
Server: Apache.1.3.12 (Unix)
Connection: close
Content-Type: text/xml
Content-Length: 124
<?xml version="1.0"?>
<methodResponse>
    <params>
        <param>
            <value><double>18.24668429131</double></value>
        </param>
    </params>
</methodResponse>
```

Les fautes en XML-RPC

- Type de réponse en XML-RPC
  - En cas de problème dans le traitement d'une requête
- L'élément methodResponse contient un sous-élément <fault> à la place de <params>
- <fault> contient un seul sous-élément <value>
  - pour indiquer l'erreur survenue

```
<?xml version="1.0"?>
<methodResponse>
    <fault>
        <value><string>No such method!</string></value>
    </fault>
</methodResponse>
```

Un exemple en XML-RPC

- Création d'un serveur en Java pouvant traiter des messages XML-RPC
- Création d'un client Java pour appeler des procédures sur le serveur
- Implémentation Java utilisée
  - Le projet Apache XML-RPC de Apache XML
    - http://xml.apache.org/xmlrpc/
    - Autre lien de téléchargement des archives nécessaires
      - http://www.java2s.com/Code/Jar/o/Downloadorgapachexmlrpcjar.htm

Un exemple en XML-RPC

- Le client
  - Appel d'une fonction sum avec 2 paramètres
  - Renvoie la somme

```
import java.util.*;
import org.apache.xmlrpc.*;
public class JavaClient {
public static void main (String [] args) {
try {
XmlRpcClient server = new XmlRpcClient("http://localhost/RPC2");
Vector params = new Vector();
params.addElement(new Integer(17));
params.addElement(new Integer(13));
Object result = server.execute("sample.sum", params);
int sum = ((Integer) result).intValue();
System.out.println("The sum is: "+ sum);
 }
catch (Exception exception) { System.err.println("JavaClient: " + exception); }
 }
}
```

Un exemple en XML-RPC

- L'objet XmlRpcClient est construit avec le nom du serveur Web suivi de /RPC2
- L'appel à server.execute(…) produit l'envoi au serveur de la requête ci-après

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<methodCall>
    <methodName>sample.sum</methodName>
    <params>
        <param> <value><int>17</int></value> </param>
        <param> <value><int>13</int></value> </param>
    </params>
</methodCall>
```

Un exemple en XML-RPC

- Le serveur

```
import org.apache.xmlrpc.*;
public class JavaServer {
public Integer sum(int x, int y){ return new Integer(x+y); }
public static void main (String [] args){
try { System.out.println("Attempting to start XML-RPC Server...");
WebServer server = new WebServer(80);
server.addHandler("sample", new JavaServer());
server.start();
System.out.println("Started successfully.");
System.out.println("Accepting requests. (Halt program to stop.)");
}
catch (Exception exception){ System.err.println("JavaServer: " + exception); }
 }
}
```

Un exemple en XML-RPC

- La méthode sum est publique
- Une instance du serveur est associée à un handler accessible au client
- Pour l'appel précédent, le serveur envoie la réponse suivante

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<methodResponse>
    <params>
        <param> <value><int>30</int></value> </param>
    </params>
</methodResponse>
```

TP1: à vous de jouer

- Compiler et exécuter les classes précédentes de l'exemple
- Concevoir et réaliser le Chatroom (vu en TP0) avec un XML-RPC en Java
  - Ressources:
    - Codes du TP0
    - archives Java (fournies) correspondant à l'implémentation Apache XML-RPC 2

Références bibliographiques

- Livre manuel sur XML-RPC.
  - http://php.net/manual/en/book.xmlrpc.php
- XML-RPC Scripting (site officiel)
  - http://xmlrpc.scripting.com/