Developing Java Clients for XML Web Services

Lect. dr. Paul POCATILU
Catedra de Informatică Economică
Academia de Studii Economice, Bucureşti

Developing and using XML Web services can be done using many programming languages. In this paper are briefly presented the Java technologies for XML Web services and is provided an example of Java client for a XML Web service that performs English-Romanian and Romanian English translations and is implemented using .NET technologies.
Keywords: *XML, Web services, JWSDP, JAX-RPC.*

Servicii Web

Serviciile Web sunt aplicaţii specificate printr-un URI (Uniform Resource Locator) care pun la dispoziţia clienţilor operaţii prin intermediul interfeţei publice, acestea fiind descrise şi utilizate prin XML folosind protocolul HTTP. Pe baza descrierii serviciului Web furnizată de către server, prin intermediul unor instrumente specializate, se generează un proxy. Cererile clienţilor sunt recepţionate de proxy iar acesta le transmite către server prin intermediul protocolului HTTP. Răspunsul de la server este recepţionat de proxy sub forma XML şi este transmis clientului ca rezultat al cererii acestuia. În figura 1 sunt prezentate interacţiunile dintre client, proxy şi server.

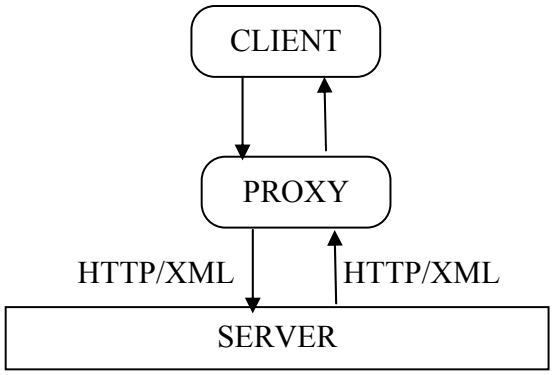

Fig.1. Accesarea serviciilor Web

Cererile suportate sunt de tipul HTTP GET, HTTP POST şi SOAP. Dezvoltarea serviciilor Web se realizează în principal utilizând Java sau platforma .NET.

Java Web Services Developer Pack (JWSDP)

Pentru realizarea de servicii Web şi clienţi Java, sunt puse la dispoziţia dezvoltatorilor mai multe tehnologii prin intermediul pachetului Java Web Services Developer Pack (JWSDP). Pachetul JWSDP conţine următoarele tehnologii care pot fi utilizate în dezvoltarea de servicii Web:

- *XML and Web Services Security* – cadru pentru securizarea serviciilor Web prin semnături digitale, criptare, utilizarea de certificate X509 etc;
- *XML Digital Signatures* – furnizează API pentru realizarea şi validarea semnăturilor electronice XML;
- *Sun Java Streaming XML Parser* – API pentru prelucrarea fişierelor XML utilizând fluxuri de date;
- *Java Architecture for XML Binding (JAXB)* – oferă suport pentru automatizarea mapării documentelor XML peste obiecte Java
- *Java API for XML-based RPC (JAX-RPC)* – API pentru realizarea de servicii Web şi clienţi Java care utilizează protocolul RPC (Remote Procedure Calls) şi XML
- *Soap with Attachments API for Java (SAAJ)* – API pentru crearea şi transmiterea de mesaje SOAP (Simple Object Access Protocol)
- *Java API for XML Registries (JAXR)* – API pentru accesarea de registrelor XML, utilizate pentru crearea, publicarea şi identificarea serviciilor Web.
- *Java API for XML Processing (JAXP)* – API pentru analiza documentelor XML
- *JavaServer Pages Standard Tag Library (JSTL)* – implementează o serie de marcatori

pentru paginile JSP

- *Java WSDP Registry Server* – sistem de stocare a informațiilor privind serviciile Web şi implementează UDDI (Universal Description, Discovery and Integration)

JAX-RPC se prezintă ca o soluție accesibilă multor dezvoltatori de aplicații distribuite, având în vedere că funcționează pe principii asemănătoare tehnologiilor CORBA sau DCOM, însă este mult mai facilă prin ascunderea unor elemente legate de protocoalele utilizate şi de maparea parametrilor.

Prin prisma tehnologiei JAX-RPC un serviciu Web este văzut ca un punct final (URL), comunicația între client şi serviciu realizându-se prin intermediul unor porturi, în funcție de protocolul de comunicație utilizat (HTTP, HTTPS, SMTP etc).

În scopul generării fişierelor proxy/stub se utilizează utilitarul JAX-RPC *wscompile*. Utilitarul *wscompile* generează următoarele categorii de fişiere:

- *stub* – clase intermediare între aplicația client şi serviciul Web; rezidă pe maşina client, furnizează clientului interfața serviciului Web şi realizează comunicarea între client şi serviciul Web;
- *tie* – clase intermediare între serviciul Web şi client; sunt stocate pe maşina server;
- *serializer* – clase generate pentru realizarea conversiei între tipurile Java şi descrierile XML şi invers;
- *WSDL* – fişiere bazate pe XML care descriu serviciile Web: numele operațiilor, URL-urile asociate, parametrii, asocierile protocoalelor cu porturile etc.

Aceste fişiere sunt utilizate de către clienții şi serviciile JAX-RPC. Generarea fişierelor se face pe baza unui fişier de configurație (document XML), furnizat ca parametru utilitarului, care precizează un fişier WSDL sau un URL către o interfață a unui serviciu compilat.

Realizarea unui client Java pentru serviciul Web *Dictionary*

În [POCA03] este prezentat un serviciu Web denumit *Dictionary* realizat utilizând tehnologia .NET şi scris în limbajul C#. Acest serviciu pune la dispoziție prin interfața sa două operații *tradu* şi *translate*. În scopul generării fişierelor necesare clientului Java prin intermediul utilitarului *wscompile* se creează fişierul de configurare *config.xml*. Conținutul acestui fişier este redat în figura 2. După cum se observă este precizată locația WSDL prin intermediul unui URL.

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration xmlns="http://java.sun.com/xml/ns/jax-rpc/ri/config">
  <wsdl location="http://paul.ase.ro/dictionary/DService.asmx?WSDL"
       packageName="DServiceStub"/>
</configuration>
```

Fig.2. Fişierul config.xml

Pentru generarea fişierelor client este utilizat instrumentul *wscompile* din linia de comandă astfel:

wscompile -gen -keep config.xml

Opțiunea *gen* are ca efect generarea fişierelor necesare clientului, iar opțiunea *keep* permite păstrarea fişierelor sursă generate.

În urma execuției utilitarului *wscompile* este creat directorul *DServiceStub*, conform numelui pachetului precizat în fişierul *config.xml* (proprietatea *packageName*) şi sunt generate fişierele din tabelul 1. După cum se observă, numărul de fişiere generate depinde de numărul de operații publicate de serviciul Web.

Tabelul 1. Fişierele generate pentru clientul Java al serviciul Web *Dictionary*

Denumire fişier	Seminificaţie
DService	Interfaţa serviciului
DService_Impl	Implementarea serviciului
DService_SerializerRegistry	Clasă pentru serializare asociată serviciului
DServiceSoap	Interfaţa SOAP a serviciului
DServiceSoap_Stub	Implementarea SOAP a serviciului
Tradu	Clase asociate operaţiei *tradu* (traduce din română în engleză) din serviciul Web *Dictionary*
Tradu_LiteralSerializer	
TraduResponse	
TraduResponse_LiteralSerializer	
Translate	Clase asociate operaţiei *translate* (traduce din engleză in română) din serviciul Web *Dictionary*
Translate_LiteralSerializer	
TranslateResponse	
TranslateResponse_LiteralSerializer	

Utilizând clasele generate este creat un client Java pentru serviciul Web *Dictionary*. Codul sursă al clientului este redat în figura 3.

```
import DServiceStub.*;
public class DServiceClient
{
 public static void main(String[] args)
 {
   if(args.length==0)
   {
      System.out.println("Utilizare:
                                  java DServiceClient <cuvint_de_tradus>");
      return;
   }
   try
   {
        DService_Impl service = new DService_Impl();
        DServiceSoap dict = service.getDServiceSoap();
        System.out.println(dict.tradu(args[0]));
    }
    catch (Exception ex)
    {
            ex.printStackTrace();
    }
 }
}
```

Fig.3. Client Java pentru utilizarea serviciului Web *Dictionary*

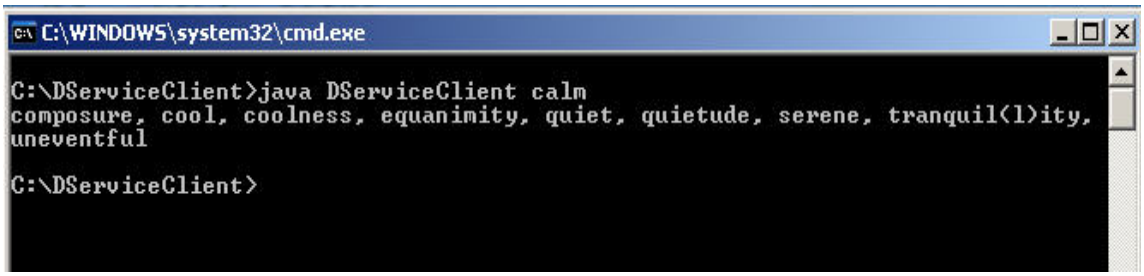

Fig.4. Exemplu de rularea a clientului Java pentru serviciul Web *Dictionary*

În figura 4 este prezentat un exemplu de rulare a clientului Java pentru serviciul Web *Dictionary*, după ce în prealabil au fost compilate fişierele.

Concluzii

Utilizarea serviciilor Web XML prezintă multiple avantaje pentru dezvoltatorii de aplicaţii precum şi pentru utilizatori. Astfel, acestea utilizează protocoale simple iar implementarea şi utilizarea acestora sunt mai uşoare decât folosind alte metode. Aplicaţiile client pot fi de tip desktop, mobile sau bazate pe Web şi astfel, poate fi utilizat orice limbaj de programare care suportă aceste protocoale, indiferent de modul în care au fost implementate serviciile Web. Se observă uşurinţa cu care pot fi realizate aplicaţii client în Java pentru serviciile Web utilizând tehnologia *JAX-RPC* indiferent de modul în care au fost realizate aceste servicii.

Bibliografie

[CHAP02] David CHAPPELL Tyler JEWELL – *Java Web Services*, O'Reilly, 2002

[POCA03] Paul POCATILU – "Dictionary Web Service", in Economy Informatics, vol. III, nr. 1, 2003, pp. 119-122

[TANA03] Ştefan TANASĂ, Cristian OLARU, Ştefan ANDREI,– *Java de la 0 la expert*, Editura Polirom, Iaşi, 2003

[TOPL03] Kim Topley – Java Web Services in Nutshell, O'Reilly, 2003

[*****] Java 2 Platform, http://java.sun.com

[*****] Java Web Services http://java.sun.com/webservices/