Il Web-Service SDMX dell'ISTAT

Esempio d'uso in applicazione web Java

Versione:	1.0.0
Data:	05/06/2014
Autore:	
Approvato da:	

Modifiche

Versione	Modifiche	Autore	Data

Indice dei contenuti

1 Introduzione

L'Istat ha implementato un WebService che sfrutta il protocollo standard SDMX (Statistical Data and Metadata eXchange) per la diffusione dei dati del corporate datawarehouse I.Stat in modalità machine-to-machine.

Il servizio offerto, attraverso i suoi metodi "esposti", potrà essere interrogato da applicazioni di qualsiasi genere, attraverso richieste standard http (GET o POST) o attraverso richieste SOAP.

Per i dettagli sui metodi esposti dal Web Service e sulle query XML necessarie alla loro interrogazione si rimanda alla "Guida all'uso del WebService SDMX"
(http://www.istat.it/it/files/2013/07/Step_funz_client_SDMXWS1.pdf)
Per approfondimenti si rimanda alla lettura della documentazione SDMX disponibile sul sito http://sdmx.org

Di seguito viene proposto un semplice esempio pratico di applicazione che "consumi" il WebService, utile per capire il funzionamento di base dello strumento.

2 Creazione dell'esempio d'uso

L'esempio è stato sviluppato con **jdk-7u65-windows-i586, apache-tomcat-7.0.55-windows-x86** utilizzando l'ambiente di sviluppo integrato **eclipse-jee-kepler-SR2-win32** e prevede una semplice applicazione web, nella quale sarà possibile sottoporre una specifica query SDMX.
In risposta l'applicazione fornirà l'output richiesto in maniera del tutto indipendente dal contesto ed in formato XML (standard SDMX Compact) sfruttando il metodo "GetCompactData" esposto dal WebService.

Per iniziare provvederemo alla creazione di nuovo progetto in **Eclipse Kepler** dal menu File- New – Dynamic Web Project.
Per iniziare verrà creata un semplice file JSP in cui inseriremo due TextBox che serviranno rispettivamente per l'input della Query XML e per la visualizzazione del relativo output.
In fine verrà creato un Button tramite cui effettueremo la chiamata al WebService.
Il passo successivo, che spiegheremo in maniera dettagliata, sarà invece quello di aggiungere al nostro progetto Java il riferimento al WebService SDMX dell'ISTAT.

2.1 Riferimento al WebService

Per aggiungere i riferimenti al web-service ISTAT nel nostro progetto sfrutteremo il tool wsimport presente nella JDK7.

Basterà quindi invocare il tool wsimport da console indicando la cartella di destinazione e l'url del webservice ISTAT:

C:\...jdk1.7\bin>wsimport -keep –s C:\ws http://sdmx.istat.it/SDMXWS/NsiStdV20Service.asmx?WSDL

Automaticamente verranno create nella cartella indicata (nel nostro esempio C:\ws), tutti i file relativi le classi per poter connettersi al webservice.

A questo punto non ci resta che copiare questi file nel nostro progetto, (nel nostro esempio in **Java Resources/src)** ed aggiungere le classi nei package appositi.

Completata tale operazione potrete vedere in "Esplora Soluzioni" di Eclipse il nuovo riferimento Web e tutti i file a lui associati (figura 2). In particolare potrete verificare la creazione della classe poxy **NSIStdV20ServiceSoap** che definisce tutti i suoi metodi con le relative funzioni per la gestione delle richieste.

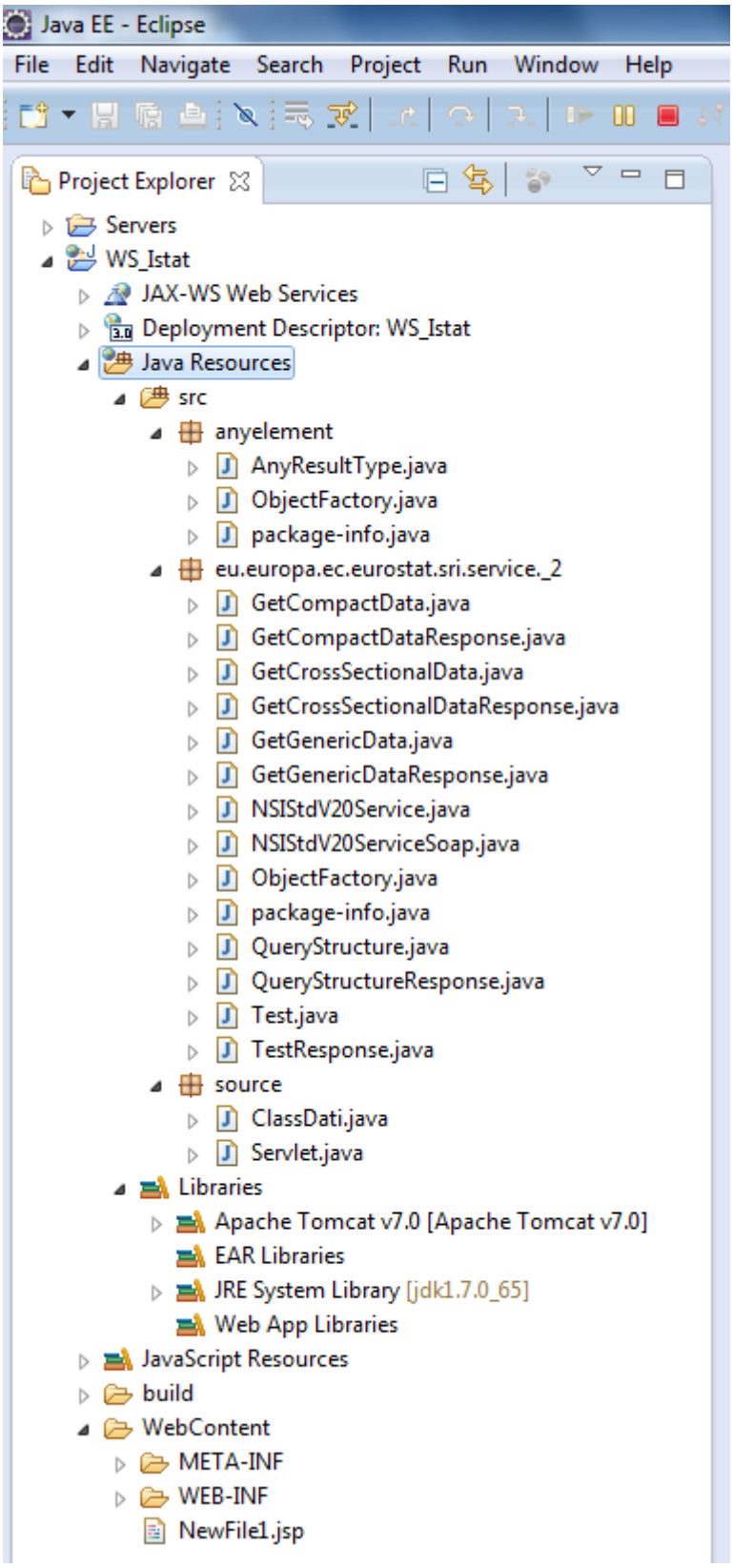

Figura 1 – Esplora soluzioni. Files creati dopo l'aggiunta del riferimento al WebService

All'interno di **NSIStdV20ServiceSoap** troveremo quindi anche il metodo **getCompactData** che è quello che noi andremo a sfruttare. Di seguito la parte di nostro interesse nel codice creato:

```
public interface NSIStdV20ServiceSoap {

@WebMethod(operationName = "GetCompactData", action =
"http://ec.europa.eu/eurostat/sri/service/2.0/GetCompactData")

@WebResult(name = "GetCompactDataResult", targetNamespace =
"http://ec.europa.eu/eurostat/sri/service/2.0")

@RequestWrapper(localName = "GetCompactData", targetNamespace =
"http://ec.europa.eu/eurostat/sri/service/2.0", className =
"eu.europa.ec.eurostat.sri.service._2.GetCompactData")

@ResponseWrapper(localName = "GetCompactDataResponse", targetNamespace =
"http://ec.europa.eu/eurostat/sri/service/2.0", className =
"eu.europa.ec.eurostat.sri.service._2.GetCompactDataResponse")

public AnyResultType getCompactData(
@WebParam(name = "Query", targetNamespace =
http://ec.europa.eu/eurostat/sri/service/2.0")
eu.europa.ec.eurostat.sri.service._2.GetCompactData.Query query);
```

Una volta creata la classe ed il riferimento al servizio non ci rimane che creare una classe che utilizzerà il metodo `getCompactData` e che sfrutteremo tramite l'evento click del controllo Button

2.2 Classe per l'invio della query SDMX

Come fatto in precedenza ci posizioniamo nella finestra "Esplora Soluzioni" e nel menù ottenuto con il tasto destro del mouse selezioniamo la voce "Aggiungi…" e poi "Classe" che nel nostro esempio chiameremo ClassDati.java.

In tale classe andremo a creare il metodo call_server(Document sdmxRequest) a cui passeremo la query XML, sotto forma di documento XML;

Al suo interno ci limiteremo ad istanziare un oggetto NSIStdV20ServiceSoap tramite il metodo getNSIStdV20ServiceSoap che farà riferimento all'endpoint da noi precedentemente indicato..

Fatto ciò sfrutteremo il suo metodo getCompactData al quale passeremo come parametro l'oggetto query, che conterrà la query in formato XML.

Di seguito riportiamo il metodo `call_server da noi creato in ClassDati`

```
public class ClassDati {

      NSIStdV20Service _NSIStdV20Service = new NSIStdV20Service();
    private AnyResultType sdmxResponse;

        public String call_server(Document sdmxRequest)
        {
              NSIStdV20ServiceSoap _NSIStdV20ServiceSoap
              =_NSIStdV20Service.getNSIStdV20ServiceSoap();

              String str = null;
              try {
                    Query query = new Query();
                    query.getContent().add(sdmxRequest.getDocumentElement());
                    sdmxResponse = _NSIStdV20ServiceSoap.getCompactData(query);
                    str = convertObjectToXml(sdmxResponse);

              } catch (Exception e) {
                    e.printStackTrace();
              }
               return str;
        }
```

Una criticità incontrata riguarda la conversione dal formato AnyResultType ritornato da getCompactData ad XML.

Per far ciò abbiamo utilizzato JAXB (API della Java Enterprise Edition) che fornisce la possibilità di serializzare oggetti Java in XML (marshalling). Di seguito il metodo convertObjectToXml che ci permette di convertire l'oggetto passato in una semplice stringa.

```
public String convertObjectToXml(AnyResultType ob)
        {
                String result = "senza risultato";
                try {
                                StringWriter sw = new StringWriter();
                                JAXBContext jaxbContext =
JAXBContext.newInstance(AnyResultType.class);
                                Marshaller jaxbMarshaller = jaxbContext.createMarshaller();

                        jaxbMarshaller.setProperty(Marshaller.JAXB_FORMATTED_OUTPUT, true);
                                jaxbMarshaller.setProperty(Marshaller.JAXB_FRAGMENT, Boolean.TRUE);

                                JAXBElement<AnyResultType> jaxbElement = new
JAXBElement<AnyResultType>(new QName(""), AnyResultType.class, ob);
                                //jaxbMarshaller.marshal(jaxbElement, file);
                                jaxbMarshaller.marshal(jaxbElement, sw);
                                result = sw.toString();

                } catch (JAXBException e) {
                                e.printStackTrace();
                }

                 return result.substring(2, result.length()-3);
        }
```

A questo punto non ci rimane che sfruttare il mettodo call_server all'interno di un Servlet.

```
if(request.getParameter("esegui") != null)
{
  ClassDati ob = new ClassDati();
  String output = null;
  String input = (String) request.getParameter("inputXML");

      if(input!=null)
      {
      Document sdmxRequest = ob.parseStringToXml(input);
      output = ob.call_server(sdmxRequest);
      }
```

3 Esecuzione applicazione

La nostra applicazione Web è completa. L'ultimo passo su cui vale la pena soffermarsi è la query SDMX che utilizzeremo in input copiandola nella casella di testo. In particolare in essa troveremo il riferimento al metodo GetCompactData e al Dataflow di nostro interesse.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetCompactData xmlns="http://ec.europa.eu/eurostat/sri/service/2.0/extended">
 <Query>
<QueryMessage ……..

  <Query>
    <query:DataWhere>
      <query:And>
        <query:Dataflow>262D_262_M</query:Dataflow>
      </query:And>
    </query:DataWhere>
  </Query>

</QueryMessage>
</Query>
</GetCompactData>
  </soap:Body>
</soap:Envelope>
```

Non ci rimane che provare quanto appena illustrato, eseguendo l'applicativo. Inseriremo nella casella in alto la query in nostro possesso, e cliccando sul bottone "Esegui…" otterremo nella casella in basso i dati richiesti.

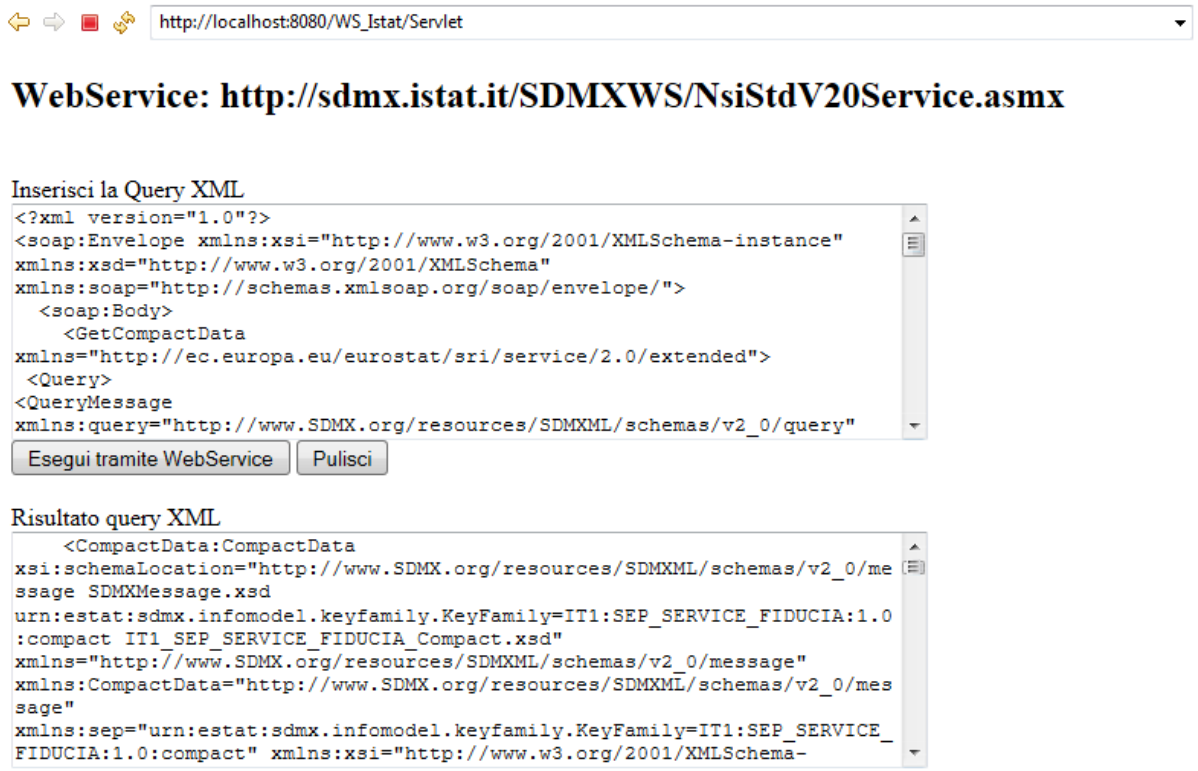

Figura 3 – Esecuzione dell'esempio d'uso