Programmazione su Reti

Esempio di applicazione Client – Server
su protocollo Soap

Fabio Donatantonio

Applicazione

Applicazione Client-Server per la gestione di un elenco di persone con relativi dati.
Viene fornito un file **xml** che funge da database, contenente i dati delle varie persone. Per ogni persona viene specificato : *codice fiscale*, *nominativo* e *lavoro svolto*.
L'applicazione lato Server fornisce una serie di opzioni per effettuare ricerche, modifiche e inserimenti.

L'applicazione utilizza due JavaBean :
- Persona
- Nominativo

Un file xlm :
- persone.xml

Una classe per la gestione del database :
- DBPersona

Una classe che definisce i metodi forniti dal Server :
- ServerPersone

Una classe di Handler per la gestione delle modifiche :
- PersonaHadler

Una classe per la definizione del Client :
- ClientPersone

Un file di deploy per l'installazione del servizio sotto Tomcat :
- deploy.wsdd

Nell'applicazione viene descritto il corretto uso dei **Bean**, la gestione dei file **xml** con **Dom**, le invocazioni di metodi tramite il protocollo **Soap**, l'utilizzo dei parametri di **IN** e **OUT** e la gestione degli intermediari (**Handler**).

Il progetto si suddivide in due package :
- *it.persona*

che contiene i JavaBean, la classe di gestione del database e la classe di Handler.

- *package di default*
  che contiene la classe Server e la classe Client, più il file di deploy.

Il package it.persona e la classe Server risiedono completamente sul server, nella cartella classes in web-inf sotto **Axis**.

Implementazione dell'applicazione

1) **FILE XML**

Persone.xml

```
<persone>
     <persona codice="FD0012">
          <nome>Fabio Donatantonio</nome>
          <lavoro>Studente</lavoro>
     </persona>
     <persona codice="MR0101">
          <nome>Marco Rossi</nome>
          <lavoro>Impiegato</lavoro>
     </persona>
     <persona codice="AB3201">
          <nome>Anna Bianchi</nome>
          <lavoro>Operaio</lavoro>
     </persona>
</persone>
```

2) **JAVA BEAN**

Persona.java

```
package it.persone;

public class Persona {
     private String codiceFiscale;
     private String nominativo;
     private String lavoro;

     public void setCodice(String c){ this.codiceFiscale=c;}
     public void setNominativo(String n){ this.nominativo=n;}
     public void setLavoro(String l)
     { this.lavoro=l;}

     public String getCodice(){ return this.codiceFiscale;}
     public String getNominativo(){ return this.nominativo;}
     public String getLavoro(){ return this.lavoro;}
}
```

Nominativo.java

```
package it.persone;

public class Nominativo {
     private String nome;

     public void setNome(String n){ this.nome=n;}

     public String getNome(){ return this.nome;}
}
```

3) CLASSE DI GESTIONE DEL DATABASE

DBPersona.java

```
package it.persone;
import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileInputStream;
import javax.servlet.ServletContext;
import javax.servlet.http.HttpServlet;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerConfigurationException;
import javax.xml.transform.TransformerException;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;

import org.apache.axis.MessageContext;
import org.apache.axis.transport.http.HTTPConstants;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;

public class DBPersona {

     Document doc;
     Persona[] lista;
     public DBPersona() throws Exception{

          MessageContext msgContext = MessageContext.
          getCurrentContext();
```

```
        HttpServlet servlet =(HttpServlet)msgContext.
        getProperty(HTTPConstants.MC_HTTP_SERVLET);
        ServletContext ctx = servlet.getServletContext();

        String file = ctx.getRealPath("persone.xml");
        BufferedInputStream in = new BufferedInputStream( new
        FileInputStream(file));
        DocumentBuilderFactory factory = DocumentBuilderFactory.
        newInstance();
        factory.setNamespaceAware(true);
        DocumentBuilder builder = factory.newDocumentBuilder();
        doc = builder.parse(in);
        NodeList pizze = doc.getElementsByTagName("persona");
        lista = new Persona[pizze.getLength()];
        for(int i=0; i<pizze.getLength(); i++){
             Element persona = (Element) pizze.item(i);
             Persona p = new Persona();
             p.setCodice(this.getAttributeValue(persona,
             "codice"));
             p.setNominativo(this.getValue(persona, "nome"));
             p.setLavoro(this.getValue(persona, "lavoro"));
             lista[i] = p;
        }
}

public int inserisciPersone(Persona[] p){
     int verifica=0;
     if(p.length==0){
          return 0;
     }
     if(p==null){
          return -1;
     }

     Element root = (Element)doc.getElementsByTagName
     ("persone").item(0);
     for(int i=0; i<p.length; i++){
          Element persona = doc.createElement("persona");
          persona.setAttribute("codice", p[i].getCodice());
          Element nome = doc.createElement("nome");
          nome.setTextContent(p[i].getNominativo());
          Element lavoro = doc.createElement("lavoro");
          lavoro.setTextContent(p[i].getLavoro());
          persona.appendChild(nome);
          persona.appendChild(lavoro);
          root.appendChild(persona);
     }

     Document exit = root.getOwnerDocument();
     DOMSource domSource = new DOMSource(exit);
     MessageContext msgContext = MessageContext.
     getCurrentContext();
```

```
        HttpServlet servlet =(HttpServlet)msgContext.getProperty
        (HTTPConstants.MC_HTTP_SERVLET);
        ServletContext ctx = servlet.getServletContext();

        File fileOut = new File(ctx.getRealPath("persone.xml"));

        StreamResult streamResult = new StreamResult(fileOut);
        TransformerFactory tf = TransformerFactory.newInstance();
        Transformer serializer;
        try {
             serializer = tf.newTransformer();
             serializer.setOutputProperty(OutputKeys.INDENT,"yes");
             serializer.transform(domSource,streamResult);
             verifica = 1;
        } catch (TransformerConfigurationException e) {
             // TODO Auto-generated catch block
             e.printStackTrace();
        } catch (TransformerException e) {
             // TODO Auto-generated catch block
             e.printStackTrace();
        }
        return verifica;
}

public Persona[] ritornaLista(){
        return this.lista;
}

public Persona personaPerCodice(String c){
        Persona p = null;
        for(int i=0; i<lista.length; i++){
             if(lista[i].getCodice().equals(c)){
                  return lista[i];
             }
        }
        return p;
}

public Persona[] personaPerLavoro(String l){
        Persona[] p=null;
        int tot=0;
        for(int i=0; i<lista.length; i++){
             if(lista[i].getLavoro().equalsIgnoreCase(l)){
                  tot++;
             }
        }
        p = new Persona[tot];
        tot = 0;
        for(int i=0; i<lista.length; i++){
             if(lista[i].getLavoro().equalsIgnoreCase(l)){
                  p[tot] = lista[i];
                  tot++;
             }
```

```
        }
        return p;
    }

    public Persona[] cercaPerNominativi(Nominativo[] nom){
        Persona[] p = null;
        int tot = 0;
        for(int i=0; i<nom.length; i++){
            Persona[] el = this.verificaNominativo
            (nom[i].getNome());
            tot = tot + el.length;
        }
        p = new Persona[tot];
        tot = 0;
        for(int i=0; i<nom.length; i++){
            Persona[] el = this.verificaNominativo
            (nom[i].getNome());
            for(int j=0 ; j<el.length; j++){
                p[tot] = el[j];
                tot++;
            }
        }
        return p;
    }

    public Persona[] verificaNominativo(String n){
        Persona[] p = null;
        int tot = 0;
        for(int i=0; i<lista.length;i++)
        {
            if(lista[i].getNominativo().toLowerCase().contains
            (n.toLowerCase()))
            {
                tot++;
            }
        }
        p= new Persona[tot];
        tot = 0;
        for(int i=0; i<lista.length; i++)
        {
            if(lista[i].getNominativo().toLowerCase().contains
            (n.toLowerCase()))
            {
                p[tot] = lista[i];
                tot++;
            }
        }
        return p;
    }

    public int cambiaLavoro(Persona p){
        int verifica=0;
```

```
            Element root = (Element)doc.getElementsByTagName
            ("persone").item(0);
            NodeList per = root.getElementsByTagName("persona");
            for(int i=0; i<per.getLength(); i++){
                  Element personael = (Element) per.item(i)
                  if(personael.getAttribute("codice").equalsIgnoreCase
                  (p.getCodice())){
                        Element lavoro = (Element) personael.
                        getElementsByTagName("lavoro").item(0);
                        lavoro.setTextContent(p.getLavoro());
                  }
            }
            Document exit = root.getOwnerDocument();
            DOMSource domSource = new DOMSource(exit);
            MessageContext msgContext = MessageContext.
            getCurrentContext();
            HttpServlet servlet =(HttpServlet)msgContext.
            getProperty(HTTPConstants.MC_HTTP_SERVLET);
            ServletContext ctx = servlet.getServletContext();

            File fileOut = new File(ctx.getRealPath("persone.xml"));

            StreamResult streamResult = new StreamResult(fileOut);
            TransformerFactory tf = TransformerFactory.newInstance();
            Transformer serializer;
            try {
                  serializer = tf.newTransformer();
                  serializer.setOutputProperty(OutputKeys.INDENT,"yes");
                  serializer.transform(domSource,streamResult);
                  verifica = 1;
            } catch (TransformerConfigurationException e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
            } catch (TransformerException e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
            }
            return verifica;
      }

      private String getValue(Element elem, String name) {
            NodeList l = elem.getElementsByTagName(name);
            if (l == null || l.getLength() == 0)
                  return null;
            elem = (Element)l.item(0);
            return elem.getFirstChild().getNodeValue();
      }

      private String getAttributeValue(Element elem,String name){
            return elem.getAttribute(name);
      }
}
```

4) **CLASSE DI HANDLER**

PersonaHandler.java

```
package it.persone;

import org.apache.axis.AxisFault;
import org.apache.axis.Constants;
import org.apache.axis.Message;
import org.apache.axis.MessageContext;
import org.apache.axis.handlers.BasicHandler;
import org.apache.axis.message.SOAPEnvelope;
import org.apache.axis.message.SOAPHeaderElement;

public class PersonaHandler extends BasicHandler{

     private static final long serialVersionUID = 1L;

     public void invoke(MessageContext msgContext) throws AxisFault
     {
          // TODO Auto-generated method stub
          boolean trovato = false;
            // Recupera l'header download
          try{
             Message reqMsg = msgContext.getRequestMessage();
             SOAPEnvelope reqEnv = reqMsg.getSOAPEnvelope();
             SOAPHeaderElement header = reqEnv.getHeaderByName
             ("http://www.unisa.it","Codicefiscale", true);

             if (header != null) {

              // se l'header esiste lo setta come processato
               header.setProcessed(true);

               // Recupera il codice fiscale
               String codice = (String)header.getValueAsType
               (Constants.SOAP_STRING);

               reqEnv.removeHeader(header);
               // Avvia la ricerca all'interno del file

               DBPersona db = new DBPersona();
               Persona p = db.personaPerCodice(codice);
               if(p==null){
                    trovato = false;
               }else{
                    trovato = true;
               }

               msgContext.setProperty("codice", new String (codice));
```

```
                msgContext.setProperty("verifica",new
                Boolean(trovato));
            }
        }catch(Exception ex){
            ex.printStackTrace();
        }
    }
}
```

5) CLASSE LATO SERVER

ServerPersone.java

```
import javax.xml.rpc.holders.IntHolder;

import org.apache.axis.MessageContext;

import it.persone.DBPersona;
import it.persone.Nominativo;
import it.persone.Persona;


public class ServerPersone {
    public Persona[] elenco() throws Exception{
        DBPersona db = new DBPersona();
        return db.ritornaLista();
    }

    public Persona cercaCodice(String code) throws Exception{
        DBPersona db = new DBPersona();
        return db.personaPerCodice(code);
    }

    public Persona[] cercaLavoro(String l) throws Exception{
        DBPersona db = new DBPersona();
        return db.personaPerLavoro(l);
    }

    public int inserisciPersone(Persona[] p) throws Exception{
        DBPersona db = new DBPersona();
        int v = db.inserisciPersone(p);
        return v;
    }

    public Persona[] cercaNominativi(Nominativo[] n,IntHolder dh)
    throws Exception{
        DBPersona db = new DBPersona();
        Persona[] p = db.cercaPerNominativi(n);
        if(p.length==0){
```

```
                dh.value=0;
                return null;
            }else{
                dh.value=p.length;
                return p;
            }
        }

        public String modificaLavoro(String lavoro) throws Exception{
            String conferma="Lavoro non modificato : ";
            MessageContext msgContext = MessageContext.
            getCurrentContext();
            boolean v = ((Boolean) msgContext.getProperty("verifica")).
            booleanValue();
            if(v==false){
                conferma = conferma+"Persona non trovata.";
            }else{
                DBPersona db = new DBPersona();
                String c = ((String)msgContext.getProperty
                ("codice")).toString();
                Persona p = db.personaPerCodice(c);
                p.setLavoro(lavoro);
                int verint = db.cambiaLavoro(p);
                if(verint ==0){
                    return conferma+"Errore nella gestione del
                    file.";
                }else{
                    conferma = "Lavoro modificato con successo.";
                }
            }
            return conferma;
        }
}
```

6) **CLASSE LATO CLIENT**

ClientPersone.java

```
import it.persone.Nominativo;
import it.persone.Persona;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.util.Map;
import javax.xml.namespace.QName;
import javax.xml.rpc.ParameterMode;
import org.apache.axis.client.Call;
import org.apache.axis.client.Service;
import org.apache.axis.encoding.XMLType;
import org.apache.axis.message.RPCElement;
```

```
import org.apache.axis.message.RPCParam;
import org.apache.axis.message.SOAPEnvelope;
import org.apache.axis.message.SOAPHeaderElement;
import org.apache.axis.soap.SOAPConstants;


public class ClientPersone {

     /**
      * @param args
      * @throws Exception
      */
     public static void main(String[] args) throws Exception {
          // TODO Auto-generated method stub

          BufferedReader in = new BufferedReader( new
          InputStreamReader(System.in) );
          String endpoint = "http://localhost:8080/axis/services
          /persone";

          while(true){
               System.out.println("1 - Elenco persone");
               System.out.println("2 - Cerca per codice fiscale");
               System.out.println("3 - Cerca per lavoro");
               System.out.println("4 - Cerca per nominativi");
               System.out.println("----------------------------");
               System.out.println("5 - Modifica lavoro persona");
               System.out.println("6 - Inserisci persone");
               System.out.println("7 - Esci");

               String scelta = in.readLine();

               if(scelta.equalsIgnoreCase("1") ||
               scelta.equalsIgnoreCase("2") ||
               scelta.equalsIgnoreCase("3") ||
               scelta.equalsIgnoreCase("4") ||
               scelta.equalsIgnoreCase("5") ||
               scelta.equalsIgnoreCase("6"))
               {

                    if(scelta.equalsIgnoreCase("1")){
                    Service service = new Service();
                    Call call = (Call) service.createCall();
                    call.setTargetEndpointAddress( new java.net.
                    URL(endpoint));

                    QName qn1=new QName("urn:BeanService","Persona");
                    call.registerTypeMapping(Persona.class, qn1,
                    new org.apache.axis.encoding.ser.
                    BeanSerializerFactory(Persona.class, qn1),
                    new org.apache.axis.encoding.
                    ser.BeanDeserializerFactory(Persona.class,qn1));
```

```
QName qn = new QName( "urn:BeanService",
"ArrayOfPersona" );
call.registerTypeMapping(Persona[].class,qn,
new org.apache.axis.encoding.ser.
BeanSerializerFactory(Persona[].class, qn),
new org.apache.axis.encoding.ser.
BeanDeserializerFactory(Persona[].class, qn));

call.setReturnType(qn);

call.setOperationName("elenco");


Persona[] app =(Persona[]) call.invoke(new
Object[]{});

if(app==null){
     System.out.println("Nessun elenco");
}else{
     for(int i=0; i<app.length; i++){
          Persona iesimo = app[i];
          System.out.println("Persona codice : "
          +iesimo.getCodice());
          System.out.println("Nome : "
          +iesimo.getNominativo());
          System.out.println("Lavoro : "
          +iesimo.getLavoro());
     }
}
}

if(scelta.equalsIgnoreCase("2")){
Service service = new Service();
Call call = (Call) service.createCall();
call.setTargetEndpointAddress( new java.net.URL
(endpoint));

QName qn1=new QName("urn:BeanService","Persona");
call.registerTypeMapping(Persona.class, qn1,
new org.apache.axis.encoding.ser.
BeanSerializerFactory(Persona.class, qn1),
new org.apache.axis.encoding.ser.
BeanDeserializerFactory(Persona.class, qn1));

call.addParameter("string", XMLType.SOAP_STRING,
ParameterMode.IN);

call.setReturnType(qn1);
call.setOperationName("cercaCodice");

System.out.println("Codice fiscale da cercare :
");
```

```
String idE = in.readLine();

Persona r = (Persona) call.invoke(new
Object[]{idE});

if(r==null){
     System.out.println("Persona non trovata.");
}else
{
          System.out.println("Persona trovata :
          ");
          System.out.println("Codice : "
          +r.getCodice());
          System.out.println("Nome : "
          +r.getNominativo());
          System.out.println("Lavoro : "
          +r.getLavoro());
}

}

if(scelta.equalsIgnoreCase("3")){
Service service = new Service();
Call call = (Call) service.createCall();
call.setTargetEndpointAddress( new java.net.URL
(endpoint));

QName qn1 = new QName( "urn:BeanService",
"Persona" );
call.registerTypeMapping(Persona.class, qn1,
new org.apache.axis.encoding.ser.
BeanSerializerFactory(Persona.class, qn1),
new org.apache.axis.encoding.ser.
BeanDeserializerFactory(Persona.class, qn1));
QName qn = new QName( "urn:BeanService",
"ArrayOfPersona" );
call.registerTypeMapping(Persona[].class, qn,
new org.apache.axis.encoding.ser.
BeanSerializerFactory(Persona[].class, qn),
new org.apache.axis.encoding.ser.
BeanDeserializerFactory(Persona[].class, qn));

call.addParameter("string", XMLType.SOAP_STRING,
ParameterMode.IN);

call.setReturnType(qn);

call.setOperationName("cercaLavoro");

System.out.println("Inserire un lavoro : ");
String idE = in.readLine();
```

```
Persona[] app = (Persona[]) call.invoke(new
Object[]{idE});

if(app==null){
     System.out.println("Nessun elenco per il
     lavoro specificato");
     }else{
     for(int i=0; i<app.length; i++){
          Persona iesimo = app[i];
          System.out.println("Persona codice : "
          +iesimo.getCodice());
          System.out.println("Nome : "
          +iesimo.getNominativo());
          System.out.println("Lavoro : "
          +iesimo.getLavoro());
          }
     }

}

if(scelta.equalsIgnoreCase("5")){
System.out.println("Inserisci codice fiscale
della persona : ");
String id = in.readLine();
System.out.println("Inserisci nuovo lavoro : ");
String l = in.readLine();

SOAPHeaderElement verificaCli = new
SOAPHeaderElement("http://www.unisa.it","Codicefi
scale");
verificaCli.setRole("http://unisa.it/roles
/modifica");
verificaCli.setValue(id);

SOAPEnvelope reqEnv = new SOAPEnvelope
(SOAPConstants.SOAP12_CONSTANTS);
reqEnv.addHeader(verificaCli);

Object [] params = new Object [] { l };
reqEnv.addBodyElement(new RPCElement("",
"modificaLavoro" , params));

Service service = new Service();
Call call = (Call) service.createCall();
call.setTargetEndpointAddress( new
java.net.URL(endpoint));
call.setSOAPVersion(SOAPConstants.SOAP12_CONSTANT
S);

SOAPEnvelope respEnv=call.invoke (reqEnv);

RPCElement respRPC = (RPCElement)respEnv.
getFirstBody();
```

```
RPCParam result = (RPCParam)respRPC.getParams()
.get(0);

String ret=((String)result.getObjectValue().
toString());

System.out.println(ret);
}

if(scelta.equalsIgnoreCase("6")){
Service service = new Service();
Call call = (Call) service.createCall();
call.setTargetEndpointAddress( new
java.net.URL(endpoint));

QName qn1 = new QName( "urn:BeanService",
"Persona");
call.registerTypeMapping(Persona.class, qn1,
new org.apache.axis.encoding.ser.
BeanSerializerFactory(Persona.class, qn1),
new org.apache.axis.encoding.ser.
BeanDeserializerFactory(Persona.class, qn1));

call.addParameter("array",qn1, ParameterMode.IN);

call.setReturnType(XMLType.SOAP_INT);

call.setOperationName("inserisciPersone");

Persona[] arr = null;

System.out.println("Quante persone vuoi inserire
? ");
int num = Integer.parseInt(in.readLine());
arr = new Persona[num];

for(int i=0; i<num; i++){
     System.out.println("Persona "+(i+1));
     System.out.println("Codice fiscale :  ");
     String code = in.readLine();
     System.out.println("Nominativo : ");
     String nome = in.readLine();
     System.out.println("Lavoro : ");
     String lavoro = in.readLine();

     Persona p = new Persona();
     p.setCodice(code);
     p.setNominativo(nome);
     p.setLavoro(lavoro);

     arr[i] = p;
}
```

```
int v = (Integer) call.invoke(new Object[]{arr});

if(v==0){
     System.out.println("Errore lunghezza
     array.");
}else{
     if(v==-1){
          System.out.println("Errore array
          nullo.");
     }else{
          System.out.println("Persone
          inserite.");
     }
}
}

if(scelta.equalsIgnoreCase("4")){
Service service = new Service();
Call call = (Call) service.createCall();
call.setTargetEndpointAddress( new
java.net.URL(endpoint));

QName qn = new QName( "urn:BeanService",
"Nominativo" );
call.registerTypeMapping(Nominativo.class, qn,
new org.apache.axis.encoding.ser.
BeanSerializerFactory(Nominativo.class, qn),
new org.apache.axis.encoding.ser.
BeanDeserializerFactory(Nominativo.class, qn));

call.addParameter("array", qn, ParameterMode.IN);

call.addParameter("quanti", XMLType.SOAP_INT,
ParameterMode.OUT);

QName qn1 = new QName( "urn:BeanService",
"Persona" );
call.registerTypeMapping(Persona.class, qn1,
new org.apache.axis.encoding.ser.
BeanSerializerFactory(Persona.class, qn1),
new org.apache.axis.encoding.ser.
BeanDeserializerFactory(Persona.class,qn1));

QName qn2 = new QName( "urn:BeanService",
"ArrayOfPersona" );
call.registerTypeMapping(Persona[].class, qn, new
org.apache.axis.encoding.ser.
BeanSerializerFactory(Persona[].class, qn),
new org.apache.axis.encoding.ser.
BeanDeserializerFactory(Persona[].class,qn);

call.setReturnType(qn2);
```

```
            call.setOperationName("cercaNominativi");

            Nominativo[] nomi = null;

            System.out.println("Quanti nominativi vuoi
            cercare ? ");
            int num = Integer.parseInt(in.readLine());
            nomi = new Nominativo[num];

            for(int i=0; i<num; i++){
                  System.out.println("Nominativo"+(i+1)+": ");
                  String n = in.readLine();
                  if(n.length()>2){
                        Nominativo nom = new Nominativo();
                        nom.setNome(n);
                        nomi[i] = nom;
                  }else{
                        System.out.println("Nominativo non a
                        ccettato. Almeno tre caratteri");
                        i--;
                  }
            }

            Persona[] p = (Persona[]) call.invoke(new
            Object[]{nomi});

            Map outputParams = call.getOutputParams();
            Integer numero = (Integer) outputParams.get(new
            QName("quanti"));

            if(p==null){
                  System.out.println("Persone trovate : "
                  +numero);
                  System.out.println("Nessuna persona trovata
                  con nominativi richiesti.");
            }else{
                  System.out.println("Persone trovate
                  "+numero+" : ");
                  for(int i=0; i<p.length; i++){
                        Persona iesimo = p[i];
                        System.out.println("Persona codice :
                        "+iesimo.getCodice());
                        System.out.println("Nome : "
                        +iesimo.getNominativo());
                        System.out.println("Lavoro : "
                        +iesimo.getLavoro());
                  }
            }
            }

      }else{
      if(scelta.equalsIgnoreCase("7")){
            System.out.println("Ciao ciao");
```

```
                    System.exit(0);
                }
            }
        }
}
}
```

7) **FILE DI DEPLOY**

Deploy.wsdd

```
<deployment xmlns="http://xml.apache.org/axis/wsdd/"
xmlns:java="http://xml.apache.org/axis/wsdd/providers/java">

<handler name="Codicefiscale" type="java:it.persone.PersonaHandler">
    <parameter name="codicef" value="polis" />
</handler>

<service name="persone" provider="java:RPC">
     <parameter name="className" value="ServerPersone"/>
     <parameter name="methodName" value="*"/>
     <role>
          http://unisa.it/roles/modifica
     </role>

     <requestFlow>
          <handler type="Codicefiscale"/>
     </requestFlow>

     <beanMapping qname="myNS:Persona" xmlns:myNS="urn:BeanService"
     languageSpecificType="java:it.persone.Persona"/>

     <arrayMapping qname="myNS:ArrayOfPersona"
     xmlns:myNS="urn:BeanService”
     languageSpecificType="java:it.persone.Persona[]"
     innerType="myNS:Persona"
     encodingStyle="http://www.w3.org/2003/05/soap-encoding"/>

     <beanMapping qname="myNS:Nominativo"
     xmlns:myNS="urn:BeanService"
     languageSpecificType="java:it.persone.Nominativo"/>

     <arrayMapping qname="myNS:ArrayOfNominativo"
     xmlns:myNS="urn:BeanService"
     languageSpecificType="java:it.persone.Nominativo[]"
     innerType="myNS:Nominativo"
     encodingStyle="http://www.w3.org/2003/05/soap-encoding"/>

     <operation name="cercaNominativi">
```

```
            <parameter name="array" mode="IN"/>
            <parameter name="quanti" mode="OUT"/>
        </operation>
    </service>
</deployment>
```