WEB SERVICE QUICK START

1. Introduzione

Per sviluppare e deployare con successo un web service sono necessarie le seguenti competenze (oltre alla conoscenza di un linguaggio di programmazione che supporti lo sviluppo di web services).

1. Buona conoscenza dei linguaggi XML e XSD, per definire correttamente il documento WSDL che descrive il funzionamento del web service.
2. Conoscenza elementare del protocollo SOAP per comprendere la struttura dei messaggi usati per invocare (ovvero *consumare*) il servizio.
3. Conoscenza elementare dell'Application Server che offrirà il servizio (ad esempio Tomcat).

Poiché il protocollo SOAP è uno degli argomenti maggiormente correlati con lo sviluppo dei web services, il presente documento tratta alcuni aspetti fondamentali del protocollo SOAP (punto 2) in riferimento alle caratteristiche di funzionamento dei web services. Per il punto 1 vedere i documenti "01_XML _quick_start.pdf", "02_XSD_quick_start.pdf" e "03_WSDL_quick_start.pdf". Il punto 3 esula dallo scopo della nostra trattazione.

2. Endpoint Reference

L'endpoint reference indica l'URL dov'è pubblicato il web service, ad esempio:

`http://localhost:8080/<`*riferimento all'Application Server*`>/services/`**foo**

L'ultima parte dell'indirizzo coincide con il nome del web service (in questo caso *foo*). Nel caso di un web service deployato su Tomcat-Axis2, ciò significa che il documento WSDL si chiamerebbe **foo.wsdl** e l'archivio avrebbe nome **foo.aar**. Il nome del web service va espresso anche all'interno della tag `name` dell'ultima sezione del documento WSDL:

```
<wsdl:service name="foo">
```

Se il web service viene esposto su Tomcat con Axis2 installato come webapp (i.e. all'interno della cartella `webapps`) il *riferimento all'Application Server* è la stringa "`axis2`". In altri casi tale riferimento potrebbe indicare il nome della web application che pubblica il servizio. Sempre nel caso della piattaforma Tomcat-Axis2, al momento di deployare il servizio occorre verificare che il file `services.xml` contenga il riferimento al nome del servizio:

```
<service name="foo">
```

L'invocazione del messaggio può avvenire in molti modi, il più diffuso è il trasporto via HTTP, il quale supporta due tipi di *binding*: SOAP e HTTP (*). In entrambi i casi il modo più semplice di invocare un metodo del servizio è quello di specificare il nome del metodo direttamente dopo l'endpoint reference:

`http://localhost:8080/<`*riferimento*`>/services/<`*nome del servizio*`>/<`*metodo*`>`

In alternativa il metodo da invocare potrebbe essere specificato dalla property `soapaction` della request, ma tale funzionalità non è sempre supportata.
Nel caso del binding HTTP il messaggio d'ingresso va appeso all'URL come una normale query string (metodo GET) oppure inviato mediante una chiamata POST. Nel binding SOAP il messaggio va indicato all'interno della tag `<body>` dell'envelope SOAP(vedi sotto).

(*) La terminologia "binding HTTP con trasporto HTTP" genera spesso parecchia confusione. Per questo motivo d'ora in poi assumeremo sempre il trasporto via HTTP, in modo da distinguere solamente tra i diversi tipi di binding.

© 2009-2011 Stefano Adriani

3. Messaggi SOAP

La struttura di un messaggio SOAP è quasi sempre la seguente:

```
<soap:Envelope>
<soap:Header>...</soap:Header>
<soap:Body>
      ...
      <soap:Fault> ... </soap:Fault>
</soap:Body>
</soap:Envelope>
```

Il blocco `Header` è opzionale, ma se presente deve essere il primo figlio della tag `<Envelope>`. Anche il blocco `Fault` è opzionale, ma se presente deve essere unico. All'interno della tag `<Envelope>` possono essere definiti i prefissi di vari namespace, esattamente come in un qualsiasi elemento *root* di un documento XML. Inoltre deve essere sempre definito il seguente namespace:

```
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
```

Un esempio di messaggio di richiesta ad un web service potrebbe essere:

```
<?xml version="1.0"?>
<SOAP-ENV:Envelope
      xmlns:q0="https://www.mywebsite.net/webservices/mynamespace/"
      xmlns:SOAP-ENV=”http://schemas.xmlsoap.org/soap/envelope/”
      xmlns:xsd=”http://www.w3.org/2001/XMLSchema”
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <SOAP-ENV:Header></SOAP-ENV:Header>
      <SOAP-ENV:Body>
      <q0:phone><number>0498978842</number></q0:phone>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

All'interno del blocco `Header` è possibile usare solamente elementi qualificati (i.e. con prefisso).
Tali elementi possono specificare i seguenti attributi:

- **mustUnderstand**: se settato ad "1" (i.e. `true`) il ricevente deve riconoscere l'elemento oppure generare un errore (un elemento `Fault`).
- **actor**: indica il nodo (vedi sotto) destinatario dell'elemento.
- **encodingStyle**: definisce i *data types* usati dall'elemento.

4. Routing

In alcuni casi un messaggio SOAP viaggia attraverso diversi "nodi" prima di giungere a destinazione dal ricevente. In questo scenario ogni nodo può avere un particolare ruolo, ad esempio trattare una parte del messaggio, eventualmente modificarlo e poi inoltrarlo. Un blocco del messaggio SOAP si dice rivolto ad un particolare nodo quando l'indirizzo del nodo corrisponde al valore dell'attributo **actor** visto sopra. In tal caso il nodo deve trattare il blocco a lui destinato e rimuoverlo dal messaggio prima di inoltrare il messaggio al nodo successivo.

© 2009-2011 Stefano Adriani

5. Esempio di Web Service

Nel paradigma dei web service la tecnologia impiegata per sviluppare il servizio dovrebbe essere trasparente all'utilizzatore finale. In altre parole il client che *consuma* il web service non deve preoccuparsi di conoscere il linguaggio utilizzato da chi pubblica il servizio: il web service deve essere fruibile da qualsiasi applicazione che interroga il servizio rispettando le specifiche del documento WSDL.
Un qualsiasi esempio concreto di sviluppo di un web service deve però fare riferimento ad una particolare tecnologia, per cui è impossibile fornire un esempio generico, senza operare scelte di campo.
Di seguito riportiamo il codice di un semplice web service realizzato in Java usando il framework **ADB di Axis2** (Axis Data Binding). Il WSDL completo del web service è riportato in appendice.

Skeleton del servizio:

Il codice Java dello skeleton (lato server del servizio) potrebbe essere:

```
public Element_out operation_name (Element_in element_in) {

        Element_in_Type element_in_Type = element_in.getElement_in() ;
        String value = element_in_Type.getElement_in_Type() ;
        Element_out_Type element_out_Type = new Element_out_Type() ;
        element_out_Type.setElement_out_Type("Ciao " + value) ;
        Element_out element_out = new Element_out() ;
        element_out.setElement_out(element_out_Type) ;
        return element_out ;
}
```

End Point Reference (SOAP):

```
http://localhost:8080/axis2/services/test/
```

End Point Reference (HTTP):

Se un servizio di nome `test` viene esposto su Tomcat-Axis2, all'interno della cartella `WEB-INF/services` di una webapp avente nome `prova`, l'endpoint dal servizio sarà qualcosa del genere:

```
http://localhost:8080/prova/services/test/operation_name
```

Esempio di request SOAP:

```
<soapenv:Envelope xmlns:q0="http://www.prova.it/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<soapenv:Body><q0:element_in>pippo</q0:element_in></soapenv:Body>
</soapenv:Envelope>
```

Esempio di response SOAP:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Body>
<element_out xmlns:ns1="http://www.prova.it/">Ciao pippo</element_out>
</soapenv:Body>
</soapenv:Envelope>
```

© 2009-2011 Stefano Adriani

Appendice

Esempio di WSDL ridotto al minimo. Questo esempio può essere usato come template per produrre un qualsiasi altro WSDL, avendo l'accortezza di mantenere consistenti gli elementi dello stesso colore.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<wsdl:definitions name="service_description"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tn="namespace_definition"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" targetNamespace="namespace_definition"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/">

  <wsdl:types>
    <xsd:schema targetNamespace="namespace_definition">
              <xsd:element name="element_in" type="xsd:string"></xsd:element>
              <xsd:element name="element_out" type="xsd:string"></xsd:element>
    </xsd:schema>
  </wsdl:types>

  <wsdl:message name="service_request">
    <wsdl:part element="tn:element_in" name="parameters"/>
  </wsdl:message>
  <wsdl:message name="service_response">
    <wsdl:part element="tn:element_out" name="parameters"/>
  </wsdl:message>

  <wsdl:portType name="port_name">
    <wsdl:operation name="operation_name">
      <wsdl:input message="tn:service_request"/>
      <wsdl:output message="tn:service_response"/>
    </wsdl:operation>
  </wsdl:portType>

  <wsdl:binding name="binding_one" type="tn:port_name">
       <soap:binding style="document"
              transport="http://schemas.xmlsoap.org/soap/http"/>
       <wsdl:operation name="operation_name">
<!-- REMARK: delete the blank space between the namespace and the operation!          -->
              <soap:operation soapAction="namespace_definition operation_name"/>
              <wsdl:input><soap:body use="literal"/></wsdl:input>
              <wsdl:output><soap:body use="literal"/></wsdl:output>
       </wsdl:operation>
  </wsdl:binding>

  <wsdl:binding name="binding_two" type="tn:port_name">
       <http:binding verb="GET"/>
       <wsdl:operation name="operation_name">
              <http:operation location="/operation_name"/>
              <wsdl:input><http:urlEncoded /></wsdl:input>
              <wsdl:output><mime:content part="element_in" type="text/xml"
/></wsdl:output>
       </wsdl:operation>
  </wsdl:binding>

  <wsdl:service name="service_name">
    <wsdl:port name="port_one" binding="tn:binding_one">
      <soap:address location="http://www.placeholder.org/"/>
    </wsdl:port>
    <wsdl:port name="port_two" binding="tn:binding_two">
      <http:address location="http://www.placeholder.org/"/>
    </wsdl:port>
  </wsdl:service>

</wsdl:definitions>
```

© 2009-2011 Stefano Adriani