Web Services Security

Introduzione ai Web Services

Davide Marrone

Sommario

- Cosa sono i web services
- Architettura dei web services
- XML-RPC
- SOAP (Simple Object Access Protocol)
- WSDL (Web Services Description Language)
- UDDI (Universal Description, Discovery and Integration)

Cos’è un Web Service?

“A Web Service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.”

Fonte: http://www.w3.org/TR/ws-arch/

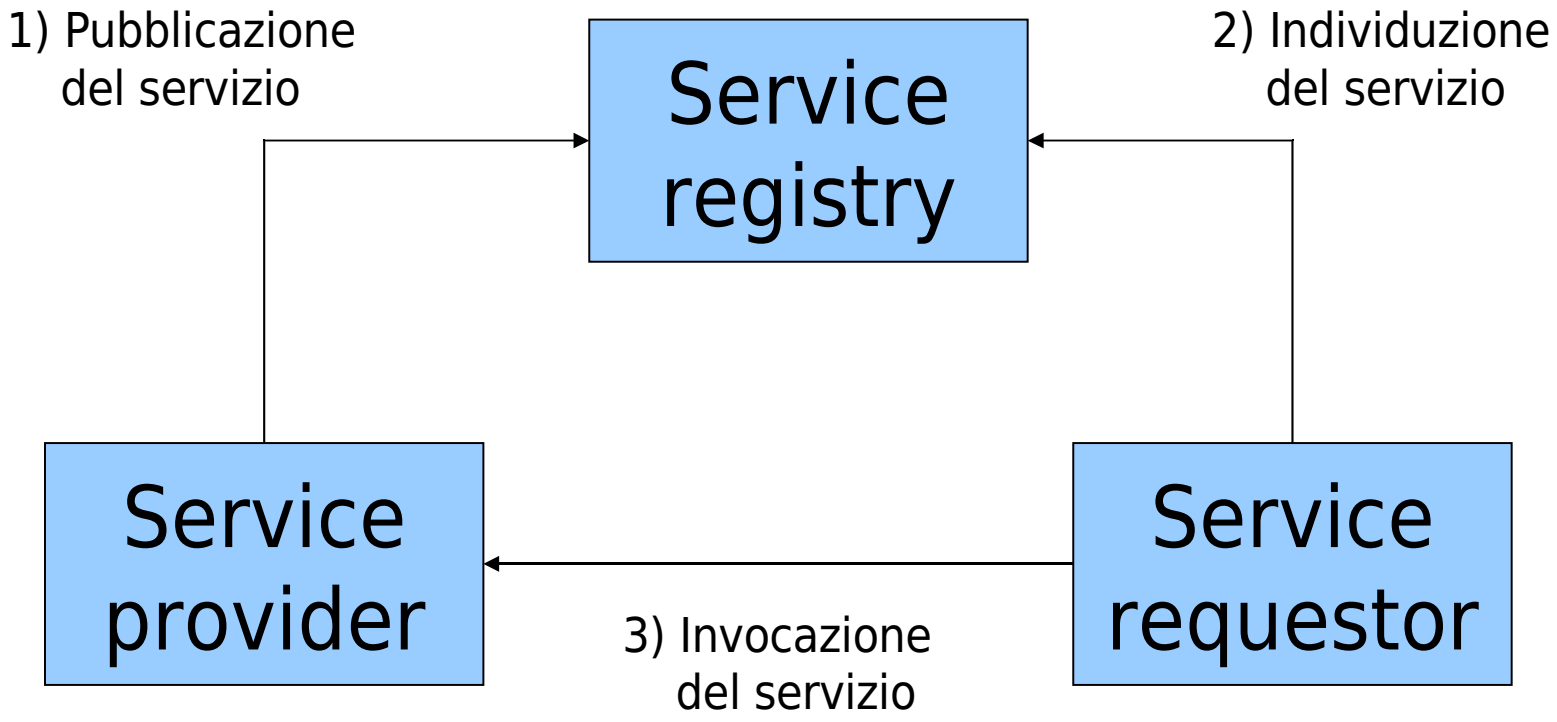
Introduzione

- Un web service è un qualsiasi servizio disponibile in rete che utilizza XML e che è indipendente da qualsiasi sistema operativo e linguaggio di programmazione
- La differenza principale tra un web service e un altro strumento utilizzato per realizzare RPC è che un web service utilizza XML per la codifica di tutti i messaggi

Proprietà dei Web Services

- Auto-descrivente: oltre alla pubblicazione del servizio dovrebbe essere pubblicata anche l'interfaccia in un formato standard
- Di facile individuazione: deve esserci un meccanismo semplice per pubblicare il servizio e per trovare la sua interfaccia

Web Services Actor



Web services protocol stack

Localizzazione	UDDI
Descrizione	WSDL
Messaggi XML	XML-RPC, SOAP, XML
Trasporto	HTTP, SMTP, FTP, BEEP

XML-RPC

- E’ un protocollo per effettuare RPC che utilizza XML e HTTP
- I messaggi XML sono contenuti nel payload delle POST HTTP
- E’ composto da tre parti:
  - XML-RPC data model
  - XML-RPC request structures
  - XML-RPC response structures

XML-RPC - Esempio

Richiesta:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<methodCall>
    <methodName>weather.getWeather</methodName>
    <params>
        <param><value>10016</value></param>
    </params>
</methodCall>
```

Risposta:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<methodResponse>
    <params>
        <param><value><int>65</int></value></param>
    </params>
</methodResponse>
```

SOAP

- E’ un protocollo basato su XML per lo scambio di informazioni tra computer
- E’ pensato per lavorare in un ambiente decentralizzato e distribuito
- Può essere utilizzato per lo scambio di messaggi e per effettuare RPC

The SOAP Message Exchange Model

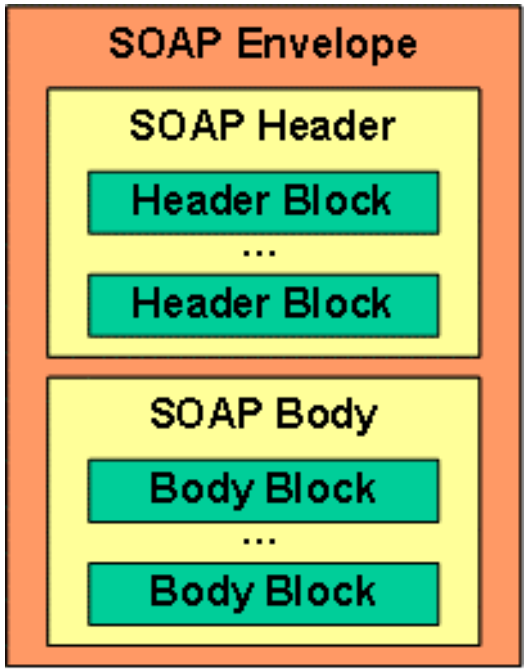
- Il modello di scambio messaggi è one-way, da un mittente ad un destinatario
  - Spesso sono usati due messaggi per permettere un modello request/response
- Le implementazione di SOAP possono sfruttare le caratteristiche del protocollo di trasporto
- I messaggi sono ruotati su quello che è chiamato “message-path”

Componenti di SOAP

- **SOAP envelope**: definisce cos'è un messaggio, chi lo deve utilizzare, quali parti sono obbligatorie e quali opzionali
- **SOAP encoding rules**: definisce le regole per l'encoding dei dati
- **SOAP RPC**: definisce una convenzione che può essere usata per effettuare RPC

SOAP Envelope

- E’ un documento XML
- Non utilizza uno schema tradizionale per il versioning
- Gli header possono essere sfruttati per estendere il messaggio es: WS-Security
- Il body è un contenitore dove sono presenti le informazioni per il destinatario del messaggio



SOAP - Esempio

```
<?xml version='1.0' encoding='UTF-8'?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2001/09/soap-envelope/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <SOAP-ENV:Body>
    <ns1:getWeather xmlns:ns1="urn:examples:weatherservice"
      SOAP-ENV:encodingStyle="http://www.w3.org/2001/09/soap-
    encoding/">
      <zipcode xsi:type="xsd:string">10016</zipcode>
    </ns1:getWeather>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

SOAP - Esempio

```
<?xml version='1.0' encoding='UTF-8'?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://www.w3.org/2001/09/soap-envelope/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <SOAP-ENV:Body>
    <ns1:getWeatherResponse
      xmlns:ns1="urn:examples:weatherservice"
      SOAP-ENV:encodingStyle="http://www.w3.org/2001/09/soap-
   encoding/">
      <return xsi:type="xsd:int">65</return>
    </ns1:getWeatherResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

SOAP Header

- **SOAP actor Attribute:** specifica se un elemento dell'header è destinato ad un particolare nodo intermedio che si trova sul message-path, il valore di questo attributo è una URI, l'actor di default è l'ultimo nodo
- **SOAP mustUnderstand Attribute:** specifica se l'actor deve processare e riconoscere correttamente la semantica di quel particolare elemento. Se il processing fallisce deve essere generato un SOAP Fault

Processing of a SOAP Message

Quando un nodo riceve un messaggio deve:

2. Identificare le parti del messaggio destinate all'applicazione
3. Verificare che le parti obbligatorie identificate al punto 1 possano essere interpretate. Se le informazioni possono essere interpretate il messaggio va processato altrimenti va scartato
4. Se il nodo non è il destinatario del messaggio deve rimuovere le parti identificate nel punto 1 prima di inoltrare il messaggio

SOAP Fault

E’ utilizzato per segnalare problemi, se presente deve essere nel body e deve essere unico

- faultcode
  - VersionMismatch
  - MustUnderstand
  - Client
  - Server
- faultstring
- faultactor
- detail

SOAP Fault - Esempio

```
HTTP/1.1 500 Internal Server Error
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn

<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
     <SOAP-ENV:Body>
          <SOAP-ENV:Fault>
               <faultcode>SOAP-ENV:Server</faultcode>
               <faultstring>Server Error</faultstring>
               <detail>
                    <e:myfaultdetails xmlns:e="Some-URI">
                       <message>My application didn't work</message>
                       <errorcode>1001</errorcode>
                    </e:myfaultdetails>
               </detail>
          </SOAP-ENV:Fault>
     </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

WSDL

- E’ un linguaggio XML che descrive web services
- Specifica la locazione dove è possibile trovare un web service e il modo in cui deve essere invocato
- Permettono di richiamare servizio in automatico
- Ci sono due sezioni
  - Definizioni astratte e descrizioni reali

WSDL – Definizioni astratte

- **types**: vengono definiti i tipi di dati che sono scambiati
- **message**: è una definizione astratta dell'informazione che viene trasmessa, è composto da uno o più elementi *part* che sono associati ad un tipo di dati (parametri)
- **portType**: è un insieme di operazioni astratte, ogni operazione fa riferimento ad un messaggio di input e uno di output (funzioni)

WSDL – Descrizioni reali

- **binding:** specifica il protocollo da utilizzare per l'operazione e il messaggio definiti in un *portType*
- **port:** specifica l'indirizzo per un binding
- **service:** è usato per raccogliere un insieme di *port* che hanno un comportamento semantico equivalente

WSDL - Esempio

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="WeatherService"
    targetNamespace="http://www.example.com/wsdl/WeatherService.wsdl"
    xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns:tns="http://www.example.com/wsdl/WeatherService.wsdl"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <message name="getWeatherRequest">
        <part name="zipcode" type="xsd:string"/>
    </message>
    <message name="getWeatherResponse">
        <part name="temperature" type="xsd:int"/>
    </message>
    <portType name="Weather_PortType">
        <operation name="getWeather">
            <input message="tns:getWeatherRequest"/>
            <output message="tns:getWeatherResponse"/>
        </operation>
    </portType>
```

WSDL - Esempio

```
<binding name="Weather_Binding" type="tns:Weather_PortType">
  <soap:binding style="rpc“ transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="getWeather">
        <soap:operation soapAction=""/>
        <input>
          <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:examples:weatherservice“ use="encoded"/>
        </input>
        <output>
          <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn:examples:weatherservice“ use="encoded"/>
        </output>
  </operation>
 </binding>
 <service name="Weather_Service">
  <documentation>WSDL File for Weather Service</documentation>
  <port binding="tns:Weather_Binding" name="Weather_Port">
      <soap:address location="http://www.example.com/weatherservice"/>
  </port>
 </service>
</definitions>
```

UDDI

- Progetto inizialmente creato da Microsoft IBM e Ariba, si divide in due parti:
  - UDDI specification: è una specifica tecnica che include informazioni su come memorizzare i dati, cercare dati esistenti e pubblicarne di nuovi
  - UDDI Business Registry: è un'implementazione di UDDI specification, lanciata da IBM e Microsoft; consente a tutti di registrare i propri servizi

Categorie UDDI

- White pages: qui sono disponibili informazioni generali riguardanti le imprese
- Yellow pages: c’è una classificazione delle aziende in base ai servizi offerti
- Green pages: informazioni tecniche sui web services con puntatori ad indirizzi per poterli invocare

UDDI - Esempio

Richiesta:

```
<find_business generic="1.0“ xmlns="urn:uddi-org:api">
                <name>Microsoft</name>
</find_business>
```

Risposta:

```
<businessList generic="1.0" operator="Microsoft Corporation“ truncated="false"
 xmlns="urn:uddi-org:api">
 <businessInfos>
  <businessInfo businessKey="0076B468-EB27-42E5-AC09-9955CFF462A3">
   <name>Microsoft Corporation</name>
    <description xml:lang="en">
          Empowering people through great software -
          any time, any place and on any device is Microsoft's
          vision. As the worldwide leader in software for personal
          and business computing, we strive to produce innovative
          products and services that meet our customer's
    </description>
```

UDDI - Esempio

```
<serviceInfos>
     <serviceInfo businessKey="0076B468-EB27-42E5-AC09-9955CFF462A3"
       serviceKey="1FFE1F71-2AF3-45FB-B788-09AF7FF151A4">
      <name>Web services for smart searching</name>
     </serviceInfo>
    <serviceInfo businessKey="0076B468-EB27-42E5-AC09-9955CFF462A3"
       serviceKey="8BF2F51F-8ED4-43FE-B665-38D8205D1333">
      <name>Electronic Business Integration Services</name>
    </serviceInfo>
    <serviceInfo businessKey="0076B468-EB27-42E5-AC09-9955CFF462A3"
       serviceKey="611C5867-384E-4FFD-B49C-28F93A7B4F9B">
        <name>Volume Licensing Select Program</name>
    </serviceInfo>
    <serviceInfo businessKey="0076B468-EB27-42E5-AC09-9955CFF462A3"
       serviceKey="A8E4999A-21A3-47FA-802E-EE50A88B266F">
        <name>UDDI Web Sites</name>
    </serviceInfo>
    </serviceInfos>
  </businessInfo>
 </businessInfos>
</businessList>
```

Bibliografia

- Web Services Architecture – W3C Note (http://www.w3.org/TR/ws-arch/)
- XML-RPC Specification (http://www.xmlrpc.com/spec)
- Simple Object Access Protocol (SOAP) 1.1 – W3C Note (http://www.w3.org/TR/2000/NOTE-SOAP-20000508/)
- Web Services Description Language (WSDL) 1.1 – W3C Note (http://www.w3.org/TR/2001/NOTE-wsdl-20010315)
- www.uddi.org
- Ethan Cerami, Web Services Essentials. O'Reilly 2002