Corso INGEGNERIA DEL WEB

ESERCITAZIONE

Installazione di Axis 2

e

creazione di un semplice web service

(a cura di Comi Ing. Antonello)

Installazione di Axis 2 e creazione di un web service

Apache Axis2 è un'infrastruttura di Apache Software Foundation per creare, pubblicare e consumare Web Service in Java.

Axis2 fornisce:

1. strumenti da riga di comando per creare e usare Web Service
2. un server SOAP stand-alone
3. una Web application che ospita i Web Services all'interno di un servlet container.

Il sito ufficiale di Axis 2 è: http://ws.apache.org/axis2/

Prerequisiti

L'uso di Axis ed il corretto funzionamento dei webservices non possono prescindere dalla presenza dell'sdk Java installato sul sistema. Pertanto prima di installare Axis2 e Tomcat assicurarsi si aver installato e correttamente configurato sulla propria macchina il jdk java. Eventualmente procedere come segue:

1. Installare il jdk (1.5 o superiore)
2. Impostare la variabile d'ambiente JAVA_HOME in modo tale che punti al pathname in cui è installato il JDK. La variabile d'ambiente JAVA_HOME può essere impostata nel file .bashrc presente all'interno della propria home directory. Nel caso in cui tale file non dovesse essere presente può essere creato direttamente dall'utente in */home/miaHome* (dove ovviamente a *miaHome* va sostituita la propria home)

Suggerimento: prima di impostare JAVA_HOME lanciare il comando

$which javac.

which ritorna il pathnames del file (o i links) che dovrebbero essere eseguiti nell'ambiente corrente. Si noti che dal pathname ottenuto dovrà essere eliminata l'indicazione della cartella bin (che verrà aggiunta in automatico al momento della chiamata a javac da parte di Tomcat). Esempio se il risultato di $ which javac è /usr/bin/javac allora nel file .bashrc scriveremo:

```
JAVA_HOME=/usr
export JAVA_HOME
```

Installazione di Tomcat

"*Apache Tomcat (o semplicemente Tomcat) è un contenitore servlet open source sviluppato dalla Apache Software Foundation. Implementa le specifiche JavaServer Pages (JSP) e Servlet di Sun Microsystems, fornendo quindi una piattaforma software per l'esecuzione di applicazioni Web*

sviluppate in linguaggio Java. La sua distribuzione standard include anche le funzionalità di web server tradizionale, che corrispondono al prodotto Apache." (http://it.wikipedia.org/wiki/Apache_Tomca)
Tomcat è il servlet containers che fornisce un ambiente per l'esecuzione di axis2 come web application rilasciata in formato WAR. Per installare Tomcat procedere come segue:

1. Collegarsi al sito ufficiale di tomcat http://tomcat.apache.org/
2. Selezionare e scaricare la *Binary Distributions*
3. Scompattare il contenuto dell'archivio in (es.) */opt/apache-tomcat-7.0.34*
4. Impostare la variabile d'ambiente *CATALINA_HOME=/opt/apache-tomcat-7.0.34 (export CATALINA_HOME)* nel file /home/miaHome/.bashrc ricordarsi di fare un *source* al termine
5. Avviare il server con il seguente comando:
   $ sh $CATALINA_HOME/bin/catalina.sh run
6. Testare tomcat (nella barra degli indirizzi del browser scrivere: localhost:8080 o nel caso di problemi usare l'IP 127.0.0.1 o ancora quello della macchina al posto di localhost)

In questa fase, dopo il test, è possibile stoppare il server.

Installare axis2 in formato WAR (Web Archive)

1. Collegarsi al sito http://axis.apache.org/axis2/java/core/download.cgi
2. Scaricare la release di Axis2 in formato WAR contenuta in un archivio ZIP
3. Estrarre il file **axis2.war** dall'archivio axis2-1.6.2-war.zip
4. Copiare il file appena estratto (axis2.war) in */opt/apache-tomcat-7.0.34/webapps*

Installare axis 2 (binary distribution)

L'installazione di Axis2 in formato binary distribution è necessaria per utilizzare gli script non inclusi in quella WAR

1. Scarichiamo l'ultima versione di axis in formato Binary Distribution dall'indirizzo: http://axis.apache.org/axis2/java/core/download.cgi
1. Scompattare il contenuto del file scaricato (Axis2 Standard Binary Distribution) in una cartella (es. /opt/axis2-1.6.2)
2. Impostare la variabile AXIS2_HOME in modo tale che si riferisca alla cartella contenente la versione di Axis 2 appena installato:
   Aprire il file /home/miahome/.bashrc ed aggiungere le seguenti righe:
   AXIS2_HOME=/opt/axis2-1.6.2

export AXIS2_HOME
(In alternativa, in Linux, per settare la variabile d'ambiente AXIS2_HOME è possibile avviare lo script setenv.sh contenuta in AXIS2_HOME/bin)

3. Impostare, nel file .bashrc (o ad esempio .bash_profile), la variabile CLASSPATH: export CLASSPATH=${CLASSPATH}:/opt/axis2-1.6.2/lib/*
4. IMPORTANTE: ogni volta che si modifica il contenuto di .bashrc (o di un file che normalmente viene letto una sola volta all'avvio della sessione di lavoro) è necessario forzare il sistema affinché ne rilegga il contenuto. Per fare ciò è sufficiente lanciare il comando $ source .bashrc. (oppure in alternativa riavviare la sessione di lavoro)

Testare Axis2

Per testare l'installazione dell'intero sistema è sufficiente usare un browser e tentare di collegarsi al servizio, pertanto:

1. Riavviare Tomcat
2. Avviare il browser e nella barra degli indirizzi scrivere: localhost:8080/axis2. Se tutto è stato effettuato in maniera corretta apparirà una schermata con tre link

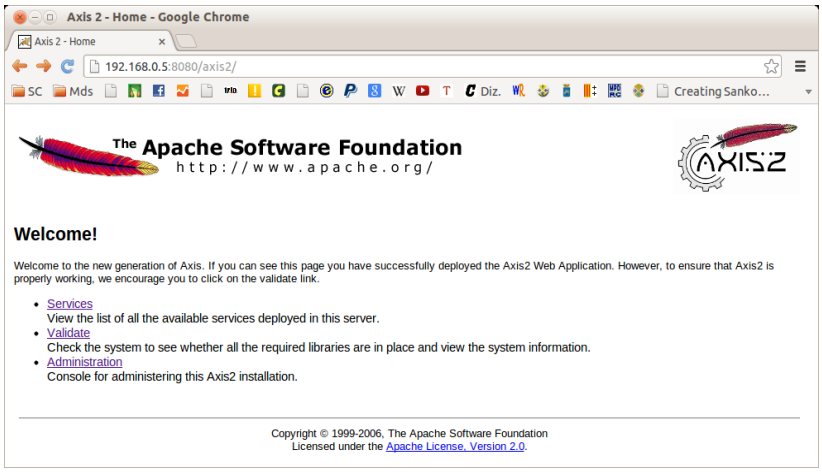

Fig. 1: La schermata di Axis2 in Tomcat

Cliccando sul link *Administration* il sistema richiederà l'immissione di un *username* e di una *password* che per default al primo accesso sono le seguenti:

username=admin
password=axis2

È possibile modificare tali info contenute in chiaro nel file WEB-INF\conf\axis2.xml.

Creazione di un web service

1. Creare la cartella "*ciaoMondo*" ed in essa la cartella "*server*"

2. All'interno della cartella "*server*" creare la cartella "*META-INF*"
3. All'interno della cartella "*server*" creare la cartella "*ciaoMondo*" che ospiterà la classe java
4. Creare la classe java "CiaoMondoService.java" e salvarla all'interno della sottocartella "*helloWorld*"

```
package ciaoMondo;
public class CiaoMondoService
{
	public String saluta(String name) {
		System.out.println("Chiamata al servizio Ciao Mondo");
		return "Ciao : " + name;
	}
}
```

Nota: l'indirizzo del package parte dalla cartella superiore escluso "server"

5. Scrivere il file "services.xml" come segue:

```
<service>
	<parameter name="ServiceClass" locked="false">ciaoMondo.CiaoMondoService</parameter>
	<operation name="saluta">
		<messageReceiver class="org.apache.axis2.rpc.receivers.RPCMessageReceiver"/>
	</operation>
</service>
```

e posizionarlo all'interno della sotto-cartella META-INF

Nel file xml di descrizione del servizio elenchiamo le classi che contengono i metodi che vogliamo pubblicare. Nel nostro esempio l'unico metodo che s'intende pubblicare è "*ciaoMondo.CiaoMondoService*". A ciascun metodo viene associata una classe che si occupa di gestire la risposta al client. In questo caso si è utilizzato la classe *apache.axis2.rpc.receivers.RPCMessageReceiver* di Axis ma è possibile scrivere anche una classe personalizzata.

6. Compilare la classe java con il comando: $ javac CiaoMondoService.java

Pubblicazione del servizio

La pubblicazione del servizio appena implementato può avvenire seguendo due strade:

1. Posizionare la cartella "*ciaoMondo*" contenuta nella cartella "*server*" all'interno della cartella "*/opt/apache-tomcat-7.0.34/webapps/axis2/WEB-INF/services*" (Ovviamente è sufficiente copiare solo i file .class)

Nota: Nel caso si incontrassero problemi si consiglia di riavviare il server tomcat! Il server andrà necessariamente riavviato ogni qualvolta si produce una modifica nel file services.xml o ad esempio si modificano le intestazioni dei metodi contenuti nel codice relativo al servizio, o ancora quando si

aggiunge nuovi metodi da pubblicare.

2. In una shell Usare il comando:

   $ jar cvf CiaoMondoService.aar *

   (In alternativa è anche possibile creare un file zip e cambiare l'estensione in .aar).

   Il file .aar dovrà contenere la cartella CiaoMondo e la cartella META-INF.

   Infine posizionare il file ottenuto in:

   /opt/apache-tomcat-7.0.34/webapps/axis2/WEB-INF/services

   oppure usare dall'interfaccia di amministrazione il tool upload file

Il servizio appena pubblicato può essere testato semplicemente osservando i servizi disponibili dall'interfaccia di amministrazione di axis2. Da un browser accedere a http://127.0.0.1:8080/axis2 quindi cliccare su *Administration*, effettuare il login ed in *System Components* selezionare il link → *Available Services.*

Se tutto si è svolto correttamente nell'elenco dei servizi sarà visibile anche quello appena pubblicato.

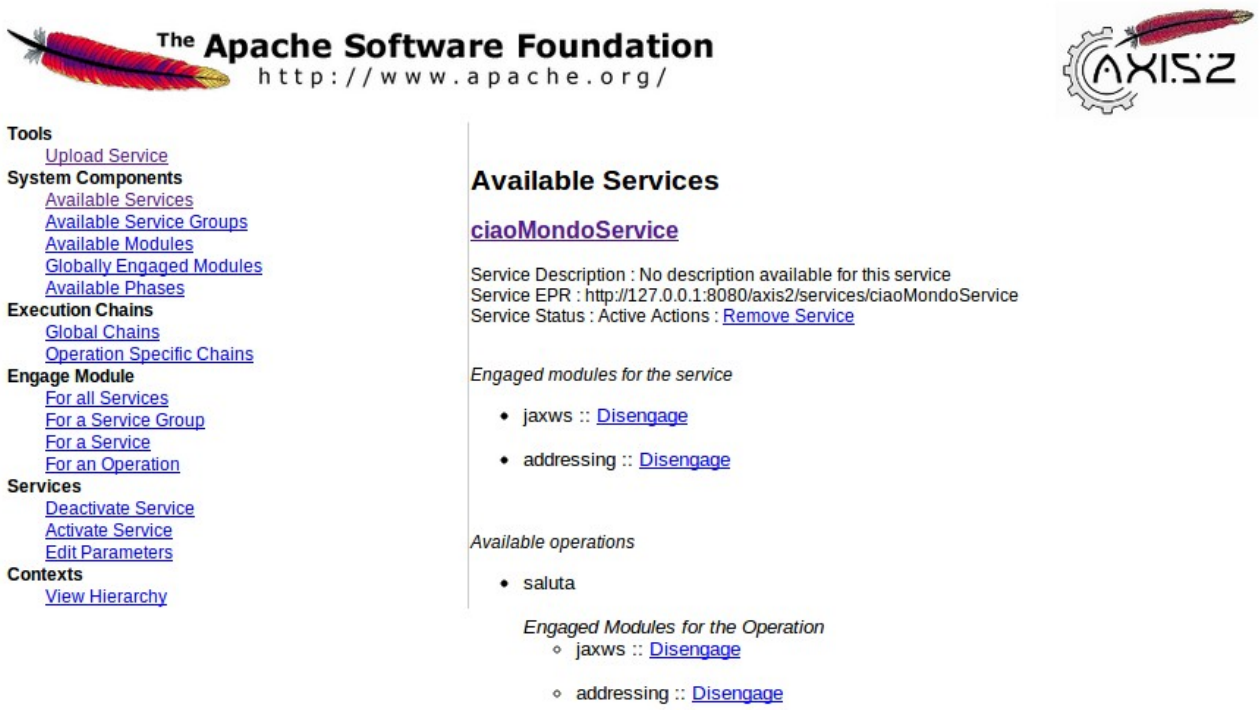

Fig. 2: La pagina relativa ai servizi pubblicati

Sviluppo del client per accedere al servizio web "Ciao Mondo".

1. Creare una nuova directory "*client*" nella cartella "*ciaoMondo*"
2. All'interno di un terminale posizionarsi all'interno della directory appena creata

3. Generare il codice del client usando il tool “wsdl2java.bat”:

   $ sudo JAVA_HOME=/usr sh $AXIS2_HOME/bin/wsdl2java.sh -uri

   http://127.0.0.1:8080/axis2/services/ciaoMondoService?wsdl -o client

Questo script, partendo dal wsdl (WEB Services Description Language) automaticamente generato, crea due file java:

a) *ciaoMondoStub.java*

b) *ciaoMondoCallbackHandler.java*

I file saranno creati all'interno della cartella *ciaoMondo/client/client/src/helloworld*

Nota: poiché i file sono stati creati con i diritti di root potrebbe essere utile lanciare il comando:

$ sudo chown -R mioAccount *

dove *mioAccount* va sostituito con il proprio nome utente.

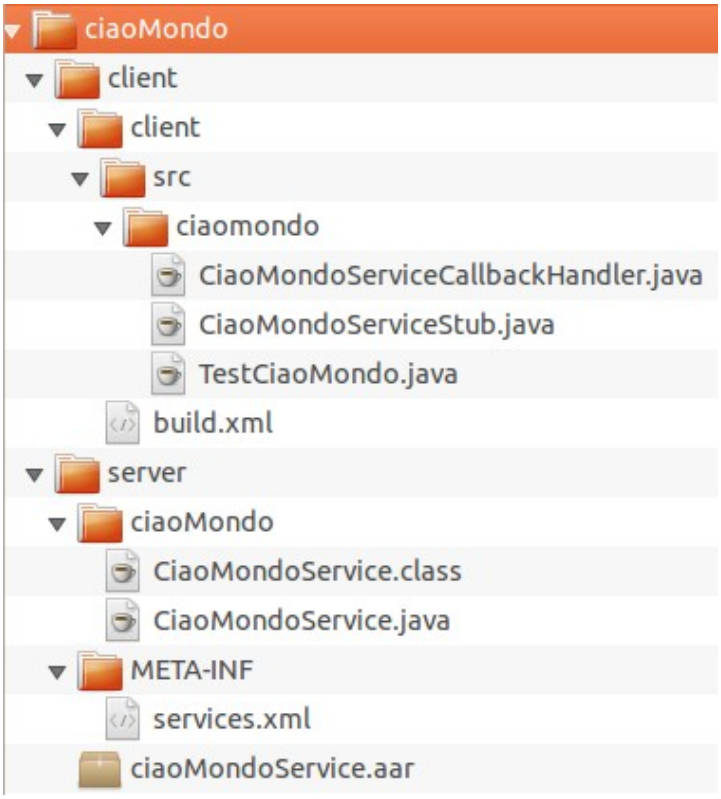

Fig. 3: La struttura dell'intero progetto

Sviluppare il codice che invoca il servizio

Creare il file “*TestCiaoMondo.java*” nella dir e salvarla in “*ciaoMondo\client\client\src\ciaomondo*”

Il codice di TestCiaoMondo.java è il seguente:

```
package ciaomondo;
import ciaomondo.*;
import ciaomondo.CiaoMondoServiceStub.Saluta;
```

```
public class TestCiaoMondo {
 public static void main(String[] args) throws Exception {
 CiaoMondoServiceStub stub = new CiaoMondoServiceStub();
 //Crea la richiesta
  ciaomondo.CiaoMondoServiceStub.Saluta  request = new ciaomondo.CiaoMondoServiceStub.Saluta();
  request.setArgs0("Pinco Pallino");
 //Invoca il servizio
  ciaomondo.CiaoMondoServiceStub.SalutaResponse response   = stub.saluta(request);
  System.out.println("Response : " + response.get_return());
  }
}
```

Compilare e testare il Web service

Spostarsi nella cartella src e compilare il nuovo file con il comando javac

$ javac -Xlint:unchecked ciaomondo/*.java

Nota: Fare attenzione che sia settata la variabile d'ambiente $CLASSPATH in .bash_profile (o .bashrc). In sostanza ci devono essere le due seguenti righe:

```
export PATH=${PATH}:/opt/android-sdk-linux/tools:/usr/bin/java:/opt/axis2-1.6.2/bin
export CLASSPATH=${CLASSPATH}:/opt/axis2-1.6.2/lib/*
```

A questo punto è possibile concludere il tutto testando il servizio. Per fare ciò lanciare il programma appena compilato con il seguente comando:

$ java ciaomondo/TestCiaoMondo

l'output sarà il seguente:

```
acomi@cassiopeaIII:~/Scrivania/ciaoMondo/client/client/src$ java ciaomondo/TestCiaoMondo
log4j:WARN No appenders could be found for logger (org.apache.axis2.description.AxisOperation).
log4j:WARN Please initialize the log4j system properly.
Response : Ciao : Pinco Pallino
```

Fig. 4: Output del comando TestCiaoMondo