Capa Integración

Configuración de Axis2 y creación de un web service

Configuración de Axis2 en Apache Tomcat

Axis2 nos provee la capacidad de agregar Web Service a las aplicaciones web y además puede funcionar como servidor autónomo.

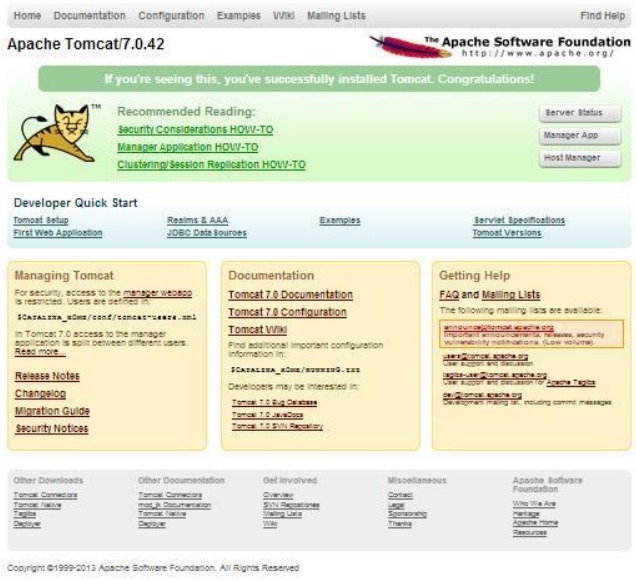
A continuación explicaremos paso a paso como configurar Axis2 en Apache Tomcat:

1. Debemos tener el servidor web Apache Tomcat que podremos descargar desde http://tomcat.apache.org/download-60.cgi , en su versión *Binary Distributions* | *Core* | *zip*. Simplemente descomprimimos el Tomcat en una carpeta, y establecemos la variable CATALINA_HOME a la ruta absoluta del directorio descomprimido. Por ejemplo, CATALINA_HOME=C:\TutorialWS\apache-tomcat-6.0.18 y añadimos a la variable PATH del sistema la ruta: C:\TutorialWS\apache-tomcat-6.0.18\bin.

2. Debemos tener el archivo WAR y Binario de axis2, los podemos encontrar en http://axis.apache.org/axis2/java/core/download.cgi. Bajar la versión más actualizada, en este caso 1.6.2.

3. Una vez descargados, deberán descomprimir los archivos. Descomprimimos la distribución binaria en una carpeta, y establecemos la variable AXIS2_HOME a la ruta absoluta del directorio descomprimido:
   Por ejemplo, AXIS2_HOME=C:\TutorialWS\axis2-1.4.1 y añadimos a la variable PATH del sistema la ruta: C:\TutorialWS\axis2-1.4.1\bin

4. Asumiendo que ya tenemos corriendo el servicio de Apache Tomcat, lo que debemos hacer es abrir un navegador y escribir la ip de nuestro equipo: el puerto donde lo configuramos o escribir localhost: puerto configurado. En este caso se ha dejado por defecto: localhost:8080. Deberíamos acceder a un página como la siguiente:

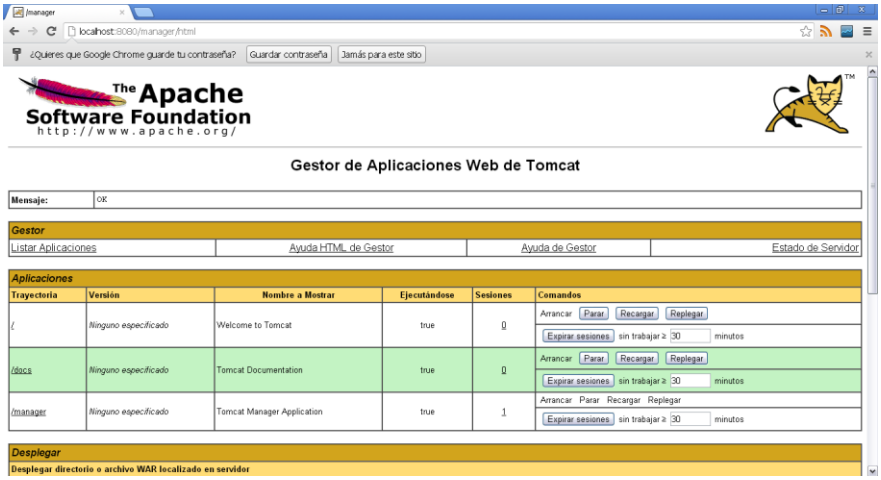


5. Luego hacemos clic en Manager App, aqui nos pedirá nuestro usuario y contraseña de tomcat, ingresar si lo conocen, en caso contrario ir al directorio de Tomcat a la carpeta conf (**C:\Archivos de programa\Apache Software Foundation\Apache Tomcat 7.0.27\conf**) y abrir el archivo "**tomcat-users.xml**".

```
<tomcat-users>
 <user name="admin" password="1234" roles="manager-gui" />
<!--
```

En caso de no encontrar estas lineas, deberán agregarlas y guardar.
Si todo a resultado bien ya podremos entrar en la configuración de Tomcat.

6. En la parte inferior de la página encontraremos una sección llamada “Desplegar “, luego “Seleccionar archivo” es aquí donde debemos seleccionar el archivo “axis2.war” que ya descargamos anteriormente (paso 1)



7. Si todo resulto bien ya tenemos configurado nuestro servidor de Apache Tomcat con Axis2 y podemos verificarlo en http://localhost:8080/axis2.

Creación de un Web Service con Axis 2

Crear el web service servidor

1. Creamos un nuevo proyecto Java en Eclipse (File | New | Java | Java Project) llamado ServidorCalculadoraWS.

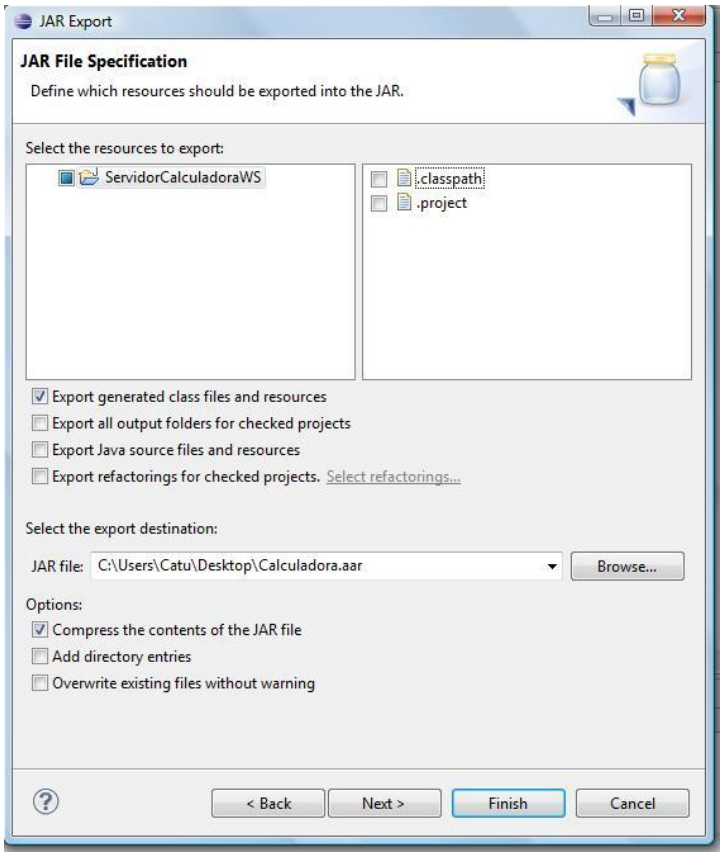
2. Ahora creamos un paquete com.ws.servidor y ahí una clase llamada Calculadora, que tiene un método que realiza la suma de dos números enteros, que recibe por parámetro y retorna el valor de la suma. También podría tener las operaciones resta, multiplicación, división, etc.

```
package com.ws.servidor;

public class Calculadora {

    /**
     * Realiza la suma dos números enteros
     * @param x Primer operando
     * @param y Segundo operando
     * @return Devuelve el resultado de la operación (x+y)
     */
    public int suma(int x, int y) {
            return x + y;
    }
}
```

3. A nivel raíz del proyecto creamos una carpeta META-INF y en su interior un fichero llamado services.xml, que será el descriptor del web service:

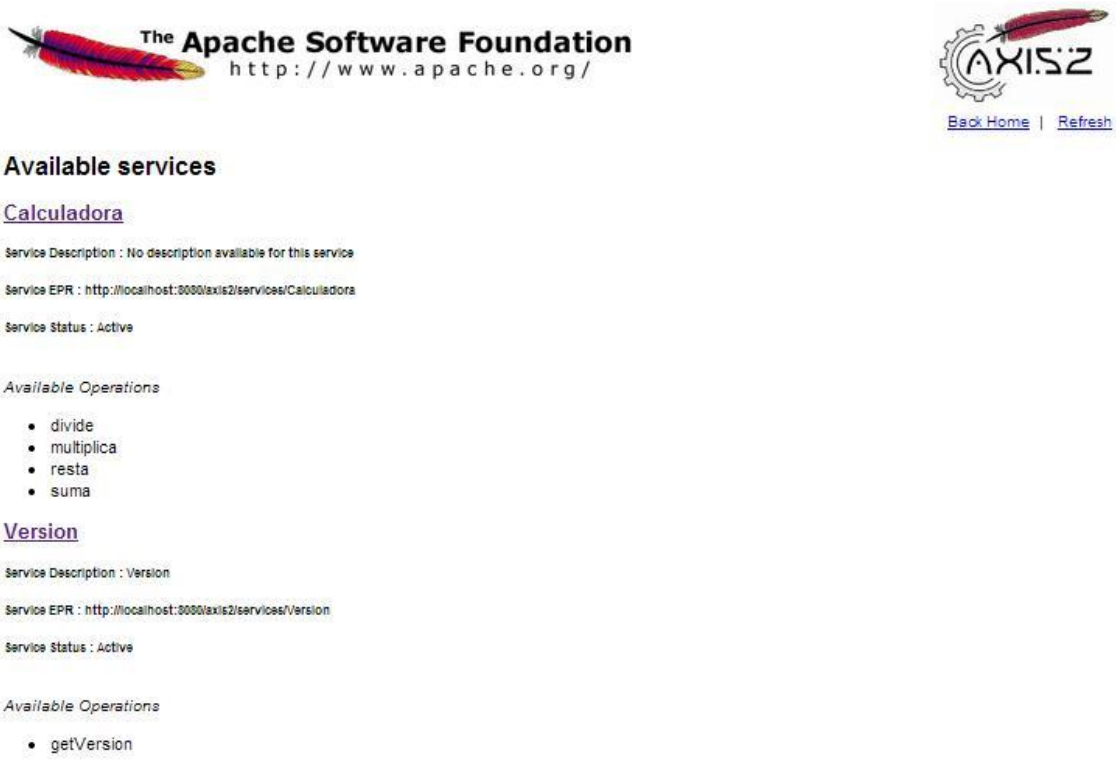
```
<service>
    <description> Web service que funciona como una calculadora
    </description>
    <messageReceivers>
        <messageReceiver mep="http://www.w3.org/2004/08/wsdl/in-out"
            class="org.apache.axis2.rpc.receivers.RPCMessageReceiver" />
    </messageReceivers>
    <parameter name="ServiceClass">com.ws.servidor.Calculadora
    </parameter>
    <operation name="suma" mep="http://www.w3.org/2004/08/wsdl/in-out" />
</service>
```

4. Vamos a crear el ensamblado de tipo aar (Axis ARrchive, que viene a ser el equivalente al jar de Java). Botón derecho sobre el nombre del proyecto | Export | Java | Jar File, pulsamos Next y en la siguiente pantalla deseleccionamos los ".classpath" y ".project". y como ruta en input JAR File ponemos por ejemplo C:\Users\Catu\Desktop\Calculadora.aar



5. Pulsamos Finish. A continuación copiamos el fichero Calculadora.aar en CATALINA_HOME\webapps\axis2\WEB-INF\services. A los pocos segundos, Axis detectará que hay un nuevo recurso que debe ser desplegado.

6. Si accedemos a la web http://localhost:8080/axis2/ y pulsamos en *Services*, veremos que aparece el nuevo servicio en la lista:



Ya tenemos el web service proveedor publicado y a la espera de que un cliente lo invoque.

Crear el web service cliente

Como hemos explicado antes, si queremos utilizar un servicio web, tenemos que utilizar un cliente que sepa 'dialogar' con aquel. Para saber qué ofrece y cómo comunicarse, nos valdremos de un descriptor del servicio, un fichero XML que cumple la especificación WSDL

1. Pediremos a Axis que nos dé el WSDL de nuestro servicio Calculadora invocando la siguiente dirección http://localhost:8080/axis2/services/Calculadora?wsdl. Lo guardamos en un fichero con el nombre calculadora.wsdl. Tenemos que verificar que al guardarlo no se introduzcan caracteres externos al propio xml. El archivo tiene que empezar como <wsdl:definitions y cerrar con el tag correspondiente.

2. También cabe aclarar que al descargar el archivo como wsdl desde el navegador hay nombres que pasan a minúscula haciendo que el archivo este incorrecto. Para ello, verificar que por ejemplo donde dice **porttype** en el archivo descargado diga **portType** como dice en el original. Chequear cuidadosamente porque también donde se definen los binding puede ocurrir lo mismo.

3. Para verificar que el archivo está correcto podemos recurrir a la siguiente página donde tenemos un validador gratis de archivos wsdl que podemos subir: http://wsdl-analyzer.com/?referer=g

4. Una vez hecho esto si vemos que en las etiquetas "<soap:address location=" aparece un número de IP en vez de localhost, cambiar a localhost.

5. Volviendo a Eclipse, creamos un proyecto Java con el nombre ClienteCalculadoraWS e importamos las librerías de Axis2: botón derecho sobre el proyecto ClienteCalculadoraWS | Build Path | Configure Build Path | pestaña Libraries | Add External JARS, navegamos hasta el AXIS2_HOME\lib,

seleccionamos todos los .jar (aunque muchos no serán realmente necesarios), y OK.

6. Copiamos ahora el fichero calculadora.wsdl en la carpeta de nuestro proyecto de eclipse de ClienteCalculadoraWS. Abrimos una consola de línea de comandos y nos situamos en el directorio del proyecto. Una vez ahí, ejecutamos el comando:

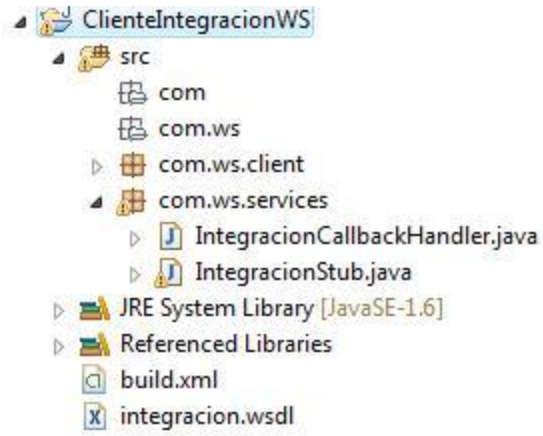
wsdl2java –uri calculadora.wsdl

7. Con ello hemos creado las clases del cliente que resuelven por nosotros la comunicación con el web service. Volvemos a Eclipse, refrescamos el proyecto y veremos que se ha creado el paquete com.ws.servidor con dos clases: CalculadoraCallbackHandler.java y CalculadoraStub.java. Creamos en el proyecto un paquete com.ws.cliente y una clase Test.java con el siguiente código:

```
import com.ws.servidor.CalculadoraStub;

import java.rmi.RemoteException;

public class Test {

      public static void main(String[] args) {

          CalculadoraStub calculadora = null;
          CalculadoraStub.Suma request = null;
          CalculadoraStub.SumaResponse response = null;

          try {

               calculadora = new CalculadoraStub();
               request = new CalculadoraStub.Suma();
               // establecemos los parámetros

               request.setSuma(7);
               request.setSuma(2);

               // invocamos al web service
               response = calculadora.suma(request);
               // mostramos la respuesta

               System.out.println("La suma es: " + response.get_return());

            } catch (RemoteException excepcionDeInvocacion) {
               System.err.println(excepcionDeInvocacion.toString());
            }


  }

}
```

Ejemplo de uso – invocación integración

En este ejemplo, se muestra como usar las clases que se tienen del lado del cliente tales como IntegracionStub e IntegracionCallbackHandler, las cuales fueron generadas a partir del archivo integracion.wsdl mediante el comando previamente explicado.



Si queremos guardar, actualizar , recuperar o eliminar un(os) objeto(os), deberá enviarse un string que represente el xml correspondiente al/los mismo(s), con el formato establecido en la documentación de Integración, e invocarse los servicios correspondientes. Si llegamos a tener un problema con el tiempo de timeout del stub, podemos modificar el mismo seteando la cantidad de milisegundos deseada.

A continuación veremos un extracto de código de cómo invocar al web service guardar datos y se muestra también como modificar el tiempo de time out del stub.

```
public static void main(String[] args) {

    IntegracionStub integracion = null;
    IntegracionStub.GuardarDatos request = null;
    IntegracionStub.GuardarDatosResponse response = null;

    try{

        integracion = new IntegracionStub();
        request = new IntegracionStub.GuardarDatos();

        Stub stub = integracion;

        stub._getServiceClient().getOptions().setTimeOutInMilliSeconds(1200000)

        String xmlUser = "<?xml version=\"1.0\"?><WS><Usuario><username>usuario

        request.setXml(xmlUser);

        response = integracion.guardarDatos(request);

        System.out.println(response.get_return());


    }catch(RemoteException excepcionDeInvocacion){
        System.err.println(excepcionDeInvocacion.toString());

    }
```

Para los casos de actualizarDatos, seleccionarDatos y eliminarDatos, la invocación es análoga. La única diferencia es que en vez de crear un objeto IntegracionStub.GuardarDatos que respresente al request y un objeto IntegracionStub.GuardarDatosResponse que represente al response, se crean los objetos IntegracionStub.ActualizarDatos e IntegracionStub.ActualizarDatosResponse, IntegracionStub.SeleccionarDatos e IntegracionStub.SeleccionarDatosResponse y IntegracionStub.EliminarDatos e IntegracionStub.EliminarDatosResponse respectivamente.