Desarrollo de Servicios Web 1

Tema Nº10:Servicios Web con AXIS2

Indicador de logro Nº10:Desarrolla servicios utilizando JAX-WS a través del framework Apache Axis2.

**TEMA 01 Teoría de los**

Imagen que contiene Icono

Descripción generada automáticamente

**TEMA Nº10:**

SERVICIOS WEB CON AXIS2

**Subtema 1:**

AXIS2

**MARCO TEÓRICO**

Apache Axis2 es un motor nuclear para [servicios web](https://es.wikipedia.org/wiki/Servicio_web). Es un rediseño total y una reimplementación completa de la ampliamente difundida pila [SOAP](https://es.wikipedia.org/wiki/SOAP) "[Apache Axis](https://es.wikipedia.org/wiki/Apache_Axis)". Existen implementaciones de Axis2 en [Java](https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)) y en [C](https://es.wikipedia.org/wiki/C_(lenguaje_de_programaci%C3%B3n)).

En el curso de la Cumbre 2004, realizada en Colombo, Sri Lanka, se introdujo una nueva arquitectura para Axis2. Esta nueva arquitectura, en la que se basa Axis2, es más flexible, eficiente y configurable en comparación con la de Axis1.x.

Ciertos conceptos probados y bien establecidos de Axis 1.x, como handlers etc., se han conservado en la nueva arquitectura.

Apache Axis2 no solo soporta SOAP 1.1 y SOAP 1.2, sino que también integra soporte para el popular estilo REST para servicios web. Una misma implementación de la lógica de negocios puede ofrecer a la vez una interfaz al estilo WS-\* como también un acceso al estilo [REST](https://es.wikipedia.org/wiki/REST)/[POX](https://es.wikipedia.org/wiki/Plain_Old_XML).

Axis2 ofrece soporte para [Spring Framework](https://es.wikipedia.org/wiki/Spring_Framework).

**Envío y recepción de mensajes**

En [Axis2](http://axis.apache.org/axis2/java/core/) presenta una forma totalmente nueva de trabajo con el XML que representa al mensaje SOAP, aunque en la superficie sea muy similar al uso de Document Oject Model (Modelo de Objetos de Documentos, DOM).

El modelo de Axis o AXIOM, se concentra en el conjunto de información del mensaje, que es la información genuina contenida en los elementos y atributos, más que en la versión en serie de las etiquetas que se ve normalmente.

Lo que es más importante, sin embargo, es que Axis2 se ocupa del mensaje SOAP por nosotros lo que nos permite concentrarnos simplemente en la creación de carga útil o, en el caso de los servicios reales en analizar la carga útil y crear una respuesta.

**¿Qué se puede hacer con Axis2?**

* Enviar mensajes SOAP
* Recibir y procesar mensajes SOAP
* Crear un servicio a partir de una clase de Java
* Crear clases de implementación para el proveedor y cliente de servicios utilizando WSDL
* Fácilmente recuperar el WSDL de un servicio
* Enviar y recibir mensajes SOAP con adjuntos
* Crear o utilizar servicios REST
* Entre otros…

**Desarrollo de Web Services con Axis2**

* Para la realización de un web services Java se utilizará Eclipse.
* Antes de empezar es necesario que tengamos instalado Apache Tomcat al Eclipse.

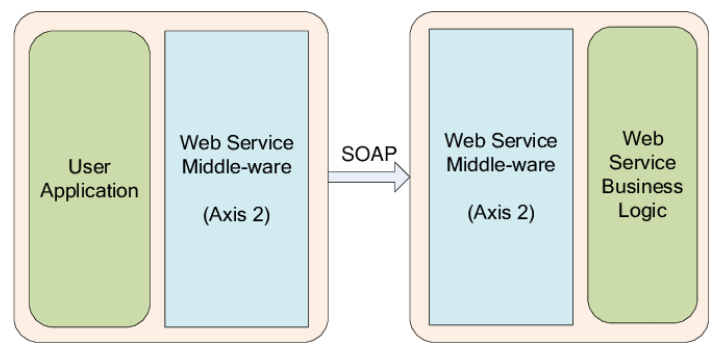
**Instalación de Apache Tomcat Server**

* Click derecho sobre Project Explorer > New > Other > Server > Next > Tomcat v7.0 Server.
* Luego le indicamos la ubicación de la instalación del servidor en nuestra máquina (Únicamente descargarlo y descomprimirlo).

Antes de continuar necesitaremos Axis2 (una implementación de SOAP). Se lo puede descargar de la siguiente dirección: <http://axis.apache.org/axis2/java/core/download.cgi>, preferiblemente descargar la última versión.

Luego de tener descargado Axis2 lo descomprimimos en el lugar que nosotros queramos en nuestra máquina. Para configurarlo en Eclipse nos ubicamos en Window > Preferences > Web Services > Axis2 > Preferences y ubicamos la ruta donde descomprimimos Axis2. Luego de ubicarla nos muestra un mensaje de que la configuración está correcta.

**Apache Axis2 Architecture**



**Librería Apache AXIS2**

La librería Apache Axis2, la última versión se descarga de la página web <http://axis.apache.org/axis2/java/core/>, una vez descargada, se descomprime los ficheros en una ruta conocida, esta ruta se usará para configurar el plugin de eclipse Axis2.



**1. MATERIALES**

Para la experiencia a realizar se requiere lo siguiente:

1. EQUIPO

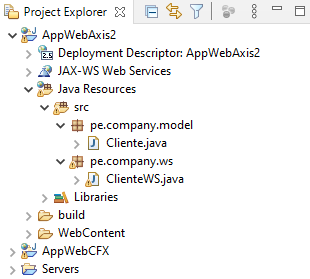
Se requiere tener un equipo de cómputo (PC o laptop).

1. HERRAMIENTAS

* Java Developer Kit (JDK) version 8.x
* Eclipse IDE
* Apache TomCat 7.x.x
* Axis2-1.6.3-bin
* SoapUI 5.5

**2. PROCEDIMIENTO**

Se tiene el proyecto Web:



Clase VO:

package pe.company.model;

public class Cliente

{

private int id\_cliente;

private String empresa;

private String vendedor;

private double credito;

public Cliente() {}

public Cliente(int id\_cliente, String empresa, String vendedor, double credito) {

this.id\_cliente = id\_cliente;

this.empresa = empresa;

this.vendedor = vendedor;

this.credito = credito;

}

//Getters and Setters

}

Clase WS:

package pe.company.ws;

import java.util.ArrayList;

import java.util.Collection;

import pe.company.model.Cliente;

public class ClienteWS

{

private static final Collection<Cliente> *itemsCliente*=new ArrayList();

static

{

Cliente cliente1=new Cliente(1021,"Acme Mfg.","Bill Adams",50000.0);

Cliente cliente2=new Cliente(1022,"Zetacorp","Larry Fitch",50000.0);

Cliente cliente3=new Cliente(1023,"Ace International","Tom Snyder",35000.0);

Cliente cliente4=new Cliente(1024,"Peter Brothers","Nancy Angelli",40000.0);

*itemsCliente*.add(cliente1);

*itemsCliente*.add(cliente2);

*itemsCliente*.add(cliente3);

*itemsCliente*.add(cliente4);

}

public ClienteWS() {

}

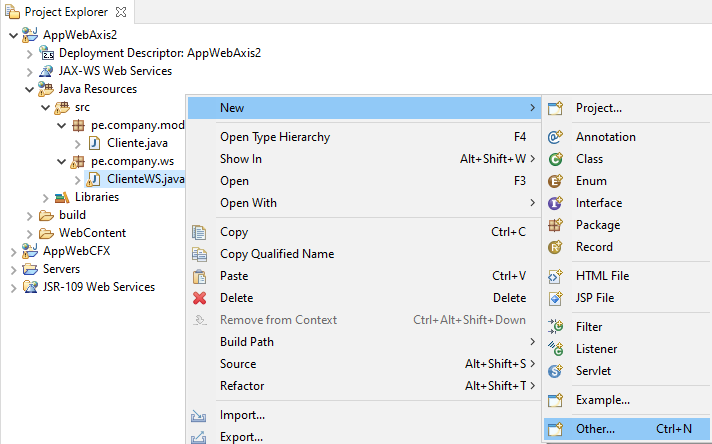
public Collection<Cliente> findAll(int codigo) {

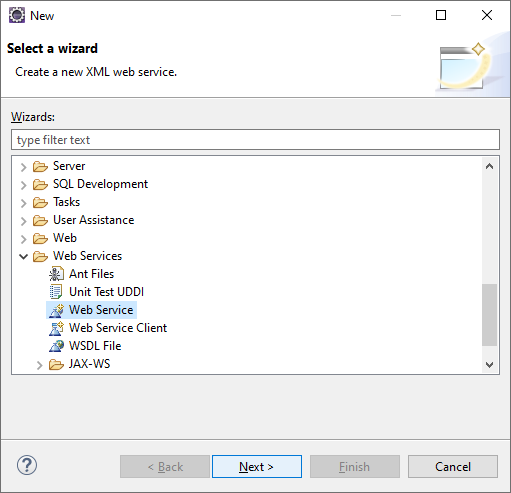
return *itemsCliente*;

}

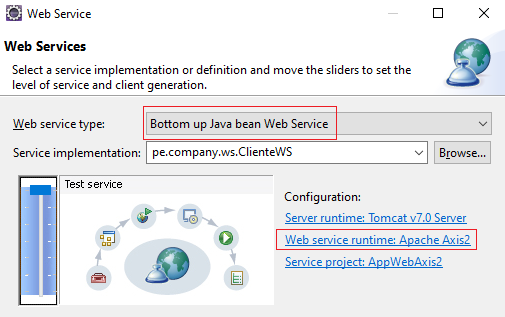
}

Ahora se pasa a crear el servicio web a partir de la clase:





Se va a indicar cuál será la implementación, presionamos en Browser y seleccionamos:

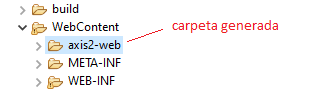


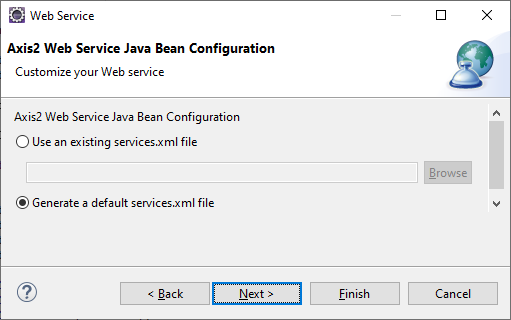
Next 🡪

Está instalando Axis2 al proyecto Web:

Nota: En caso le genere una excepción es debido que no agregó el entorno de tiempo de ejecución Axis2, debe pasar a configurar Apache Axis2 con su instructor.

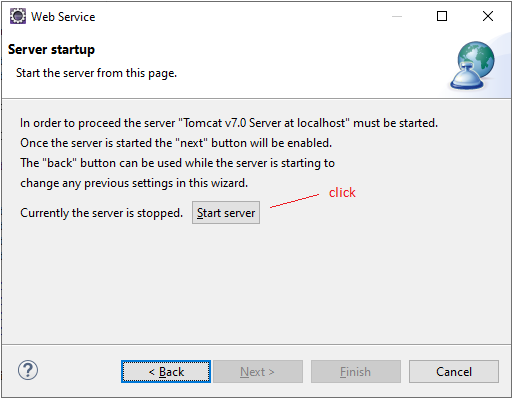
Se observa que acaba de crearse una carpeta:





Next (Generará un service.xml que tendrá toda la configuración de todos los servicios del proyecto) 🡪

Aparecerá la ventana que nos indica que debemos tener iniciado el TomCat

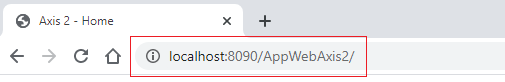


Next 🡪 Finish

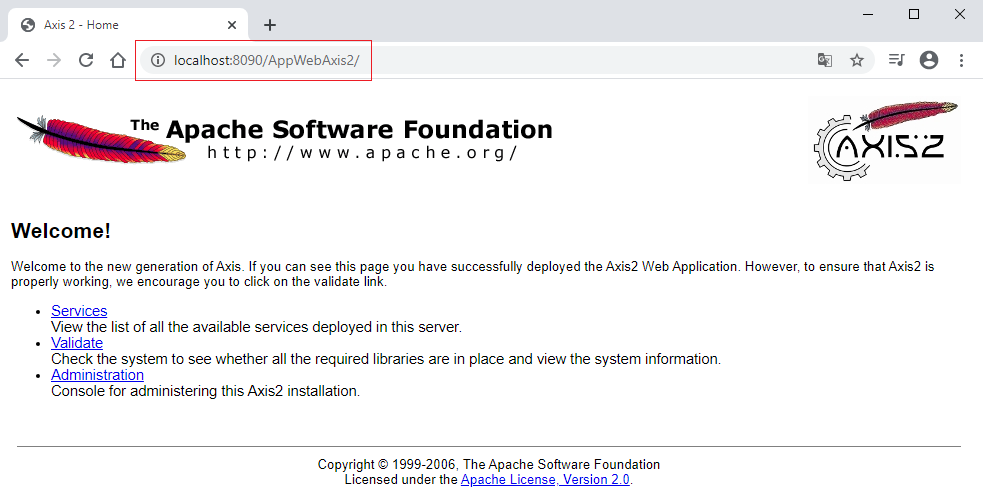
Una vez desplegado en Tomcat:



Nos dirigimos al navegador, agregamos:



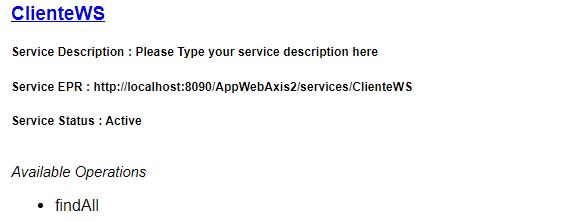
Se podrá visualizar 3 enlaces:



El enlace Services permitirá ver todos los servicios que están implementados en el proyecto:



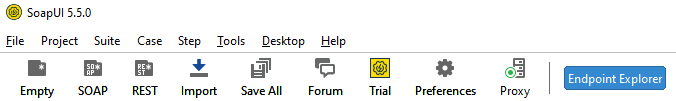
Accediendo se podrá ver el servicio implementado:

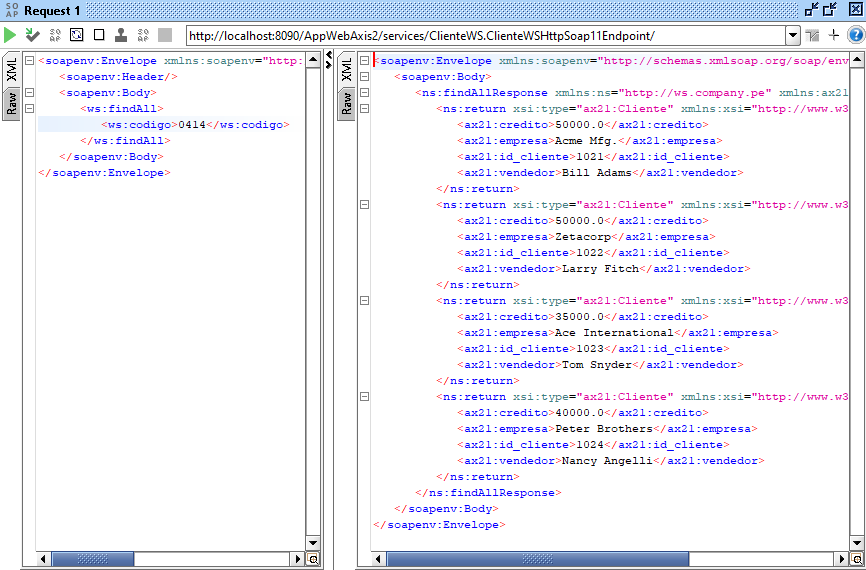


Al hacerle click se verá el WSDL:



Para verificar el Web Service se pasa hacer la prueba unitaria con el SoapUI:





**ACTIVIDAD VIRTUAL:**

1. **CUESTIONARIO TÉCNICO**

Revisa y analiza el tema desarrollado en la presente sesión, luego responde las siguientes preguntas propuestas:

* ¿Por qué usar Apache Axis2?
* ¿Es mejor Apache Axis2 sobre Apache CXF, dar una breve explicación?
* ¿Qué metodología se utiliza para generar WSDL en Axis2?
* ¿Axis2 puede integrarse con Spring mejor que CXF, dar una breve explicación?
* Mediante Eclipse IDE, Base de datos, Apache Tomcat y Apache Axis2 crear web services para las operaciones a la BD. Realizar pruebas unitarias.

1. **ENLACES DE REFERENCIA**

* https://ideca.gov.co/sites/default/files/documentacion/defestandares-recursosadopcion-de-servicios-web10.pdf
* https://www.tamps.cinvestav.mx/~vjsosa/clases/sd/README\_ejemploHolaMundo\_WebService.pdf

1. **CONCLUSIONES DE LA EXPERIENCIA**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_