Práctica 3 (1ª Parte): Desarrollo de un Servicio Web en Axis2

1. Objetivos

El objetivo de esta práctica es:

1. Practicar el desarrollo top-down de servicios web.
2. Practicar la definición de ficheros WSDL y XSD.
3. Practicar el desarrollo de clientes de un servicio web.
4. Utilizar herramientas de prueba de servicios web.

2. Actividades

1ª Parte: Desarrollo de un servicio web (Proveedor)

Suponiendo el mismo esquema de datos de la Práctica 1, desarrollar un servicio denominado `ImpuestoCirculacionSS` que proporcione la siguiente funcionalidad:

- Dado el dni de un contribuyente, el servicio retorna el valor total del impuesto de circulación a pagar por el contribuyente.
- Dado el dni de un contribuyente, el servicio retorna su nombre y apellidos y la lista de vehículos a su nombre.
- Ambas operaciones deberá retornar un error si el dni no pertenece a ningún contribuyente.

Para el desarrollo del servicio deberán seguirse los siguientes pasos:

1. Especificación.
   Formulación del documento WSDL que describe el servicio requerido.
2. Implementación.
   Desarrollar una implementación del servicio usando Axis2.
3. Empaquetado.
   Generar el archivo .aar correspondiente al servicio (`ImpuestoCirculacionSS.aar`).
4. Despliegue.
   Desplegar el servicio en Tomcat.

2ª Parte: Utilización de la herramienta SoapUI para probar el servicio

Probar el funcionamiento del servicio desarrollado utilizando la herramienta SoapUI, comprobando los mensajes de petición y respuesta en los siguientes casos:

- Invocación correcta de la operación que retorna los datos de un seguro.
- Invocación con fallo de la operación que retorna los datos de un seguro.

3ª Parte: Desarrollo de un cliente de un servicio web (Consumidor)

Utilizando Axis2, desarrollar un cliente sencillo en Java que pruebe la funcionalidad del servicio anterior en todos sus posibles escenarios. Utilizar cliente síncrono.

4ª Parte: Utilización de la herramienta TCPMon para capturar mensajes SOAP

Utilizando la herramienta TCPMon, capturar los mensajes de petición y respuesta intercambiados entre el cliente y el servicio `ImpuestoCirculacionSS` en los siguientes casos:

- Invocación correcta de una operación del servicio.
- Invocación incorrecta de una operación del servicio.

Aspectos importantes a tener en cuenta:

- En la fase de especificación del servicio:
  - En caso de que sean necesarias, las clases de dominio ya definidas en el XML Schema de la práctica 1 deben reutilizarse (no declararse de nuevo dentro del WSDL). Para ello, importar el XML Schema de la práctica 1 en el WSDL (concretamente, en el XML Schema que se define dentro de la sección types). El mecanismo para importar XML Schemas lo tenéis en las transparencias Tema 1.

  - Cuando se generan los stubs y skeletons desde un descriptor WSDL utilizando Axis2, se crean también todas las clases auxiliares JAXB que dan soporte a los tipos del esquema. En lugar de esas, usad directamente las creadas en la práctica 1 (en las que puede que ya hayáis incluido código de negocio).

  - Cuidado con los namespaces y los nombres de paquetes Java. Cómo se va a importar el XML Schema de la práctica 1 (con namespace `http://www.unican.es/ss/XXX`), es mejor usar un namespace distinto para el WSDL, por ejemplo, `http://www.unican.es/ss/ImpuestoCirculacionSS`.

- En la fase de implementación del servicio:
  - Para acceder a los datos del sistema usaremos las interfaces `IContribuyentesDAO` que tenéis disponibles en Moodle. Debéis desarrollar una clase `ConntribuyentesDAOImpl` que implemente dichas interfaces (basta con implementar los métodos que necesitéis). En un servicio real los datos a los que se accede desde dicha interfaz podrían estar almacenados en una base de datos. En este caso, usaremos documentos .xml como mecanismo de almacenamiento de datos. Por ello, la clase `ConntribuyentesDAOImpl` debe cargar los datos desde un documento .xml usando JAXB (cómo en la primera práctica).

  - La clase `AyuntamientoDAOImpl` y el fichero del que se tomen los datos se empaquetará con el resto del servicio, lo cual impide el uso de paths relativos en el acceso a ficheros, es decir una orden cómo

    ```
    new File("xmlFiles/ayuntamiento.xml");
    ```

    no funciona, pues el servicio no se está ejecutando desde el directorio raíz dónde está la carpeta xmlFiles. Buscad la solución por internet para realizar ese acceso a los datos y en caso de no encontrarla, consultad al profesor.

- En la fase de empaquetamiento del servicio:
  - El archivo .aar debe contener toda aquella información necesaria para el funcionamiento del servicio, entre ella, los ficheros .wsdl y todos los ficheros de los que éste depende. En concreto, si se ha importado un XML Schema externo, será necesario que éste se incluya en el archivo .aar.
    Para que todos los ficheros .xsd que se encuentren en la carpeta `resources` se incluyan en el archivo `.aar` hay que realizar una pequeña modificación del fichero `build.xml` generado por Axis2 al crear los Skeletons. Intentad encontrar qué modificación hace falta y en caso de duda consultar al profesor.

3. Entrega y evaluación

Esta práctica se entregará al finalizar la segunda parte (siguiente sesión de prácticas).

La documentación a entregar acerca de esta parte de la práctica es la siguiente:

- Código completo tanto del servicio como del cliente (en forma de proyectos Eclipse exportados).
- Archivo `ImpuestoCirculacionSS.aar`.

Así mismo, será necesario realizar una pequeña presentación al profesor demostrando el funcionamiento de la práctica. La presentación se realizará en la siguiente sesión de prácticas o en un horario establecido con tal propósito.