WebServices SOAP

Dirección General de Tecnologías de la Información y Comunicaciones

Áreas de Gobierno Tecnológico de SSII y del Dato

Servicio Andaluz de Salud
Consejería de Sanidad, Presidencia y Emergencias

Contenido

Resumen

- **Versión:** v01r01
- **Fecha publicación:** 18 nov 2016
- **Entrada en vigor desde:**18 nov 2016

Cumplimiento normativo

Las normas expuestas son de obligado cumplimiento. La STIC podrá estudiar los casos excepcionales los cuales serán gestionados a través de los responsables del proyecto correspondiente y autorizados por el Área de Gobernanza de la STIC. Asimismo cualquier aspecto no recogido en estas normas deberá regirse en primera instancia por las guías técnicas correspondientes al esquema nacional de seguridad y esquema nacional de interoperabilidad según correspondencia y en su defecto a los marcos normativos y de desarrollo software establecidos por la Junta de Andalucía, debiendo ser puesto de manifiesto ante la STIC.

La STIC se reserva el derecho a la modificación de la norma sin previo aviso, tras lo cual, notificará del cambio a los actores implicados para su adopción inmediata según la planificación de cada proyecto.

En el caso de que algún actor considere conveniente y/o necesario el incumplimiento de alguna de las normas y /o recomendaciones, deberá aportar previamente la correspondiente justificación fehaciente documentada de la solución alternativa propuesta, así como toda aquella documentación que le sea requerida por la STIC para proceder a su validación técnica.

Contacto Arquitectura: l-arquitectura.stic@juntadeandalucia.es

Histórico de cambios

Los cambios en la normativa vendrán acompañados de un registro de las modificaciones. De este modo se podrá realizar un seguimiento y consultar su evolución. Ordenándose de mas recientes a menos recientes, prestando especial cuidado a las cabezeras de la tablas dónde se indican las fechas de entrada en vigor y versión.

Versión	v01r01	**Fecha publicación**	18 nov 2016	**Fecha entrada en vigor**	18 nov 2016
Alcance					
• Versión inicial sobre las normas y recomendaciones sobre como crear servicios y clientes SOAP.					

1. Introducción

En el siguiente texto tiene como objetivo mostrar cómo debe generarse y usarse un servicio SOAP siguiendo la normativa JEE de la STIC. Para comprender mejor las indicaciones dadas, se ha creado un ejemplo ilustrativo partiendo del Arquetipo base "javaee7-sample" de la STIC. Vea la documentación asociada al arquetipo base para comprender cómo configurar el proyecto.

El ejemplo modificado puede descargarse de la siguiente ruta: http://git.sas.junta-andalucia.es/examples/javaee7-sample/tree/SoapIntegration

Este proyecto se centra en mostar un ejemplo de cómo crear y usar servicios web en aplicaciones Java.

En el ámbito de la STIC, **los servicios web son definidos por la Oficina Técnica de Interoperabilidad (OTI)**, por lo que tanto el servicio web como el cliente debe ser creado a partir de una definición que nos viene ya dada. Esta definición es ofrecida como un **fichero WSDL** a partir del cual crearemos nuestros servicios y clientes.
A modo de ejemplo, en este proyecto se han utilizado los siguientes servicios publicados por la OTI:
* Ejemplo de cliente SOAP: Servicio "Consulta de Informe"
* Ejemplo de servicio SOAP: Servicio "S050 - Recepción de Ingreso de usuario"

Adicionalmente, para la gestión de los mensajes enviados y recibidos se hace uso de la librería HAPI (http://hl7api.sourceforge.net/)

2. Cliente SOAP desde WSDL

Para la generación de un cliente SOAP desde un WSDL existen varias opciones que podrían utilizarse. En este caso se va a hacer uso de Maven para generar el cliente fácilmente sin necesidad de acudir a herramientas externas. De esta forma, los pasos a seguir son:

1. Incluir WSDL en nuestro proyecto: Para poder generar el Cliente SOAP debemos partir de un WSDL que nos debe ser proporcionado por la OTI. Normalmente estos WSDL se encuentran en los contratos de integración. Los pasos realizados en este ejemplo han sido.
    - Obtener archivo ConsultaInforme.wsdl del contrato de integración
    - Copiar el archivo obtenido en la ruta src/main/resources/wsdl/
2. Configurar Maven: Configurar archivo pom.xml para generar archivos Java a partir del WSDL

```
<plugin>
        <groupId>org.codehaus.mojo</groupId>
        <artifactId>jaxws-maven-plugin</artifactId>
        <version>2.4.1</version>
        <executions>
            <execution>
                <goals>
                    <goal>wsimport</goal>
                </goals>
            </execution>
        </executions>
        <configuration>
            <verbose>true</verbose>
            <!-- Input -->
            <wsdlDirectory>src/main/resources/wsdl/</wsdlDirectory>
            <wsdlFiles>
                <wsdlFile>ConsultaInforme.wsdl</wsdlFile>
            </wsdlFiles>
            <!-- Output -->
            <packageName>fromwsdl.output</packageName>
            <!-- Need for access to referenced files in wsdl -->
            <vmArgs>
                <vmArg>-Djavax.xml.accessExternalSchema=all</vmArg>
            </vmArgs>
        </configuration>
</plugin>
```

3. Ejecutar Maven usando la instruccion "jaxws:wsimport"

4. Mover código de la carpeta generada ("target/generated-sources") al paquete que se desee de nuestro proyecto.

Una vez generadas las clases puede usarse el cliente web desde cualquier punto de nuestro código, por ejemplo:

```
URL wsURL = new URL("http://servidor/ConsultaInforme?wsdl");
   ConsultaInformeService soapService = new ConsultaInformeService(wsURL);
   ConsultaInformeServiceSoap soap = soapService.getConsultaInformeServiceSoap();

   MensEntrada entrada = generarMensajeEntrada();
   MensSalida salida = soap.consultaInforme(entrada);
   // Procesar mensaje de salida
```

Probar el cliente

Para realizar una prueba del cliente generado en este proyecto de ejemplo realizaremos los siguientes pasos:

a. Abrir en un navegador la ruta "localhost:7001/javaee7-sample/faces/pages/documents/viewDocument.xhtml"
b. Abrir un simulador de Servicio Web, como por ejemplo SoapUI, y publicar un servicio en la ruta "http://localhost:9999/consultaInforme?wsdl" (Puede usar el mensaje ubicado en "/doc/DOC_T12.xml" como ejemplo de respueta válida)
c. Editar, si se desea, la ruta destino a la que se va a realizar la petición
d. Pulsar sobre "Ejecutar"
e. En los espacios "XML Petición" y "XML Resultado" se debe mostrar los mensajes generados tanto para la petición como para la respuesta.

3. Servidor SOAP desde WSDL

El proceso para generar un servicio SOAP partiendo de un WSDL es similar al de la creación de un cliente, con la salvedad de que la generación automática del cliente genera una interfaz que puede ser usada directamente y en el caso de la generación de un servicio web la interfaz debe ser implemtada con una clase, añadiendo a esta la lógica de negocio que debe ofrecer el servicio web.
De esta forma, los pasos a seguir son:

a. Generar clases del cliente SOAP siguiendo las instrucciones del apartado anterior, en este caso siguiendo el contrato del servicio S050
b. Implementar la interfaz generada o sustituir interfaz por clase conservando todas las anotaciones. En el caso de nuestro proyecto de ejemplo la interfaz generada es "S050ServiceSoap" y ha sido transformada a clase respentando la especificación.
c. Añadir a la anotación @WebService la url en la que se desea publicar el servicio. En nuestro caso deseamos que la ruta del web service tenga el siguiente formato "http://servidor:port/javaee7-sample/ws/s050". Para ello basta con modificar la anotación indicada de la siguiente forma:

```
@WebService(serviceName = "/ws/s050", ....)
    public class S050ServiceSoap {
```

Probar el servicio

Para realizar una prueba del servicio generado en este proyecto de ejemplo realizaremos los siguientes pasos:

a. En SoapUI, crear un cliente SOAP a partir del mismo wsdl que hemos usado para generar el servicio en Java.
b. Configurar mensaje que va a lanzarse al servicio creado (Puede usar el mensaje ubicado en "/doc/ADT_A01.xml" como ejemplo de mensaje válido)
c. Lanzar una petición al servicio web creado, en nuestro caso "http://localhost:7001/javaee7-sample/ws/s050"
d. En la pantalla de resultado debe mostrarse la respuesta recidiba desde el servidor.

ANTES de lanzar el mensaje puede acceder a la página http://localhost:7001/javaee7-sample/faces/pages/patients/patients.xhtml, donde verá que el paciente "AN000000001" no tiene ningún episodio.
TRAS el envío de mensaje podrá recargar esta página y verá como ahora debe tener un episodio cargado. Igualmente podrá acceder a la base de datos y verá como la tabla ARQ_EPISODE tiene un nuevo episodio asociado al paciente con ID 2