Valparaíso, 24 de abril de 2006

Manual de referencia para la invocación de WebServices con Aduanas (SMS v3.0)

Introducción

El sistema SMS (Sistema de Mensajería por SOAP) fue diseñado con el fin de servir como herramienta de apoyo a los programadores para desarrollar sistemas similares, que se encarguen de gestionar mensajería vía SOAP con los WebServices de Aduana.

Para facilitar la comprensión del sistema SMS, es necesario contar con una documentación detallada que explique el funcionamiento del sistema en sus aspectos más importantes, centrándose principalmente en la invocación de los WebServices. Este es el objetivo que pretende cubrir el presente documento.

Plataforma Software

Los ejemplos tratados en este documento se basan en:

- API SOAP V 2.0 de Apache
- Lenguaje de programación Java (Runtime 1.3.1)

Es importante destacar que nuestro ejemplo esta basado en Java, pero lo anterior *no limita la comprensión y portabilidad* a otros lenguajes, ya que SOAP precisamente fue diseñado para lograr la invocación de Servicios que pueden estar desarrollados en leguajes distintos. Por lo anterior, el demo desarrollado (SMS) es fácilmente homologable a cualquier lenguaje que utilice SOAP para invocar servicios remotos.

Por último, es importante destacar que al final de este documento, se entrega un conjunto de Links los cuales pueden ser utilizados para profundizar en los temas tratados.

Normalizaciones

Se han definido las siguientes normalizaciones en la estructura de nombre de los archivos:

1.- La estructura del nombre del archivo XML que viajará desde y hacia el servidor de Aduanas es la siguiente:

Emisor-Documento-Versión-A-Correlativo-Extra.XML

• **Emisor**: Sigla que identifica al emisor del mensaje (descripción libre). Formato: alfanumérico de largo máximo 4.

• **Documento**: Código que identifica el tipo de documento de acuerdo a la lista de códigos del ANEXO 51-45 del Compendio de Normas Aduaneras. Formato: alfanumérico de largo máximo 15.

• **Versión**: Identificación de la versión de la estructura del mensaje, por ejemplo 1.0. Formato: dos dígitos separados por un punto.

• **A**: corresponde a la acción o funcionalidad del mensaje, ejemplo I: Ingreso; M: Modificación; A: Aclaración. Formato: alfanumérico de largo 1.

• **Correlativo**: número de referencia del mensaje o el identificador dado por el emisor. Formato: numérico de largo 15.

• **Extra**: Alfanumérico del largo que permita el sistema operativo de uso libre para el usuario.

2.- El nombre de los archivos de respuesta se estructura de la siguiente forma:

respuesta-n-nombre original del mensaje.xml

• Respuesta: Es una constante (El nombre “respuesta”) que identifica al tipo de archivo.

• n: Representa un correlativo que indica el número de respuesta para el archivo original.

3.- El archivo XML con todos los tag de respuestas obtenido desde el servidor, se copia en la carpeta Terminados, con la siguiente estructura de nombre

Emisor-Documento-Versión-Número_aceptación.XML

• Número_aceptación: Es el número entregado por el servidor de Aduanas, cuando el documento cumple con todas las validaciones.

Las normalizaciones antes descritas, son utilizadas por MIDAS y el sistema SMS, para facilitar la gestión de los archivos.

Ciclo de vida de un mensaje

Las carpetas utilizadas en el sistema SMS son las siguientes:

Carpeta	Descripción
mensajes	En esta carpeta se deben ingresar los documentos XML, que cumplan la normalización de nombre descritas en el ítem "normalizaciones". Esta carpeta será consultada por el sistema SMS para verificar la estructura de los archivos XML presentes en ella, los archivos que superen la evaluación.
xml/pendientes	Una vez que los archivos XML son enviados al servidor del SNA, son trasladados de la carpeta **mensajes** a **pendientes**. Los archivos permanecerán en la carpeta **pendientes** por las siguientes razones: - El archivo enviado, todavía no se procesa. - No existe conexión con el servidor. Una vez recibida una respuesta del servidor, el archivo pendiente puede pasar a la carpeta **terminados** o **rechazados**, según sea el caso.
xml/terminados	Los archivos que reciben un mensaje de "aceptación" desde el servidor del SNA, son trasladados desde la carpeta **pendientes** a la **terminados**. SMS, recupera todos los nuevos tag de respuesta enviados por el servidor de Aduanas, estos tag son incorporados en el documento antes de ser copiado a la carpeta **terminados**. La respuesta recibida del servidor del SNA se almacena en la carpeta **respuestas.**
xml/rechazados	Si un archivo que fue enviado recibe una respuesta indicando la existencia de errores en el archivo enviado, se copia la respuesta recibida (El nombre tiene que cumplir lo indicado en normalizaciones) en la carpeta **respuestas.** A continuación se anexan tags de control en el documento original, los cuales registran los errores descritos en la respuesta del servidor del SNA.
xml/respuestas	En esta carpeta se almacenan todas las repuestas recibidas desde el servidor. Las repuestas se almacenan en archivos XML con el formato de nombres de archivos descrito en "normalizaciones".

A continuación se presenta un grafo que describe el ciclo de vida de la mensajería por SOAP:

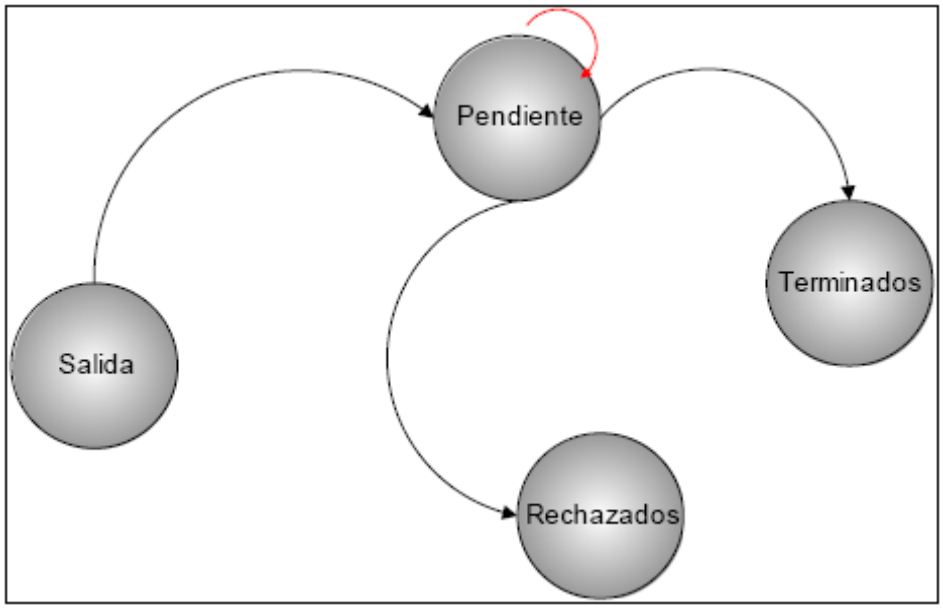

Figura 1: Grafo del ciclo de vida de los mensajes

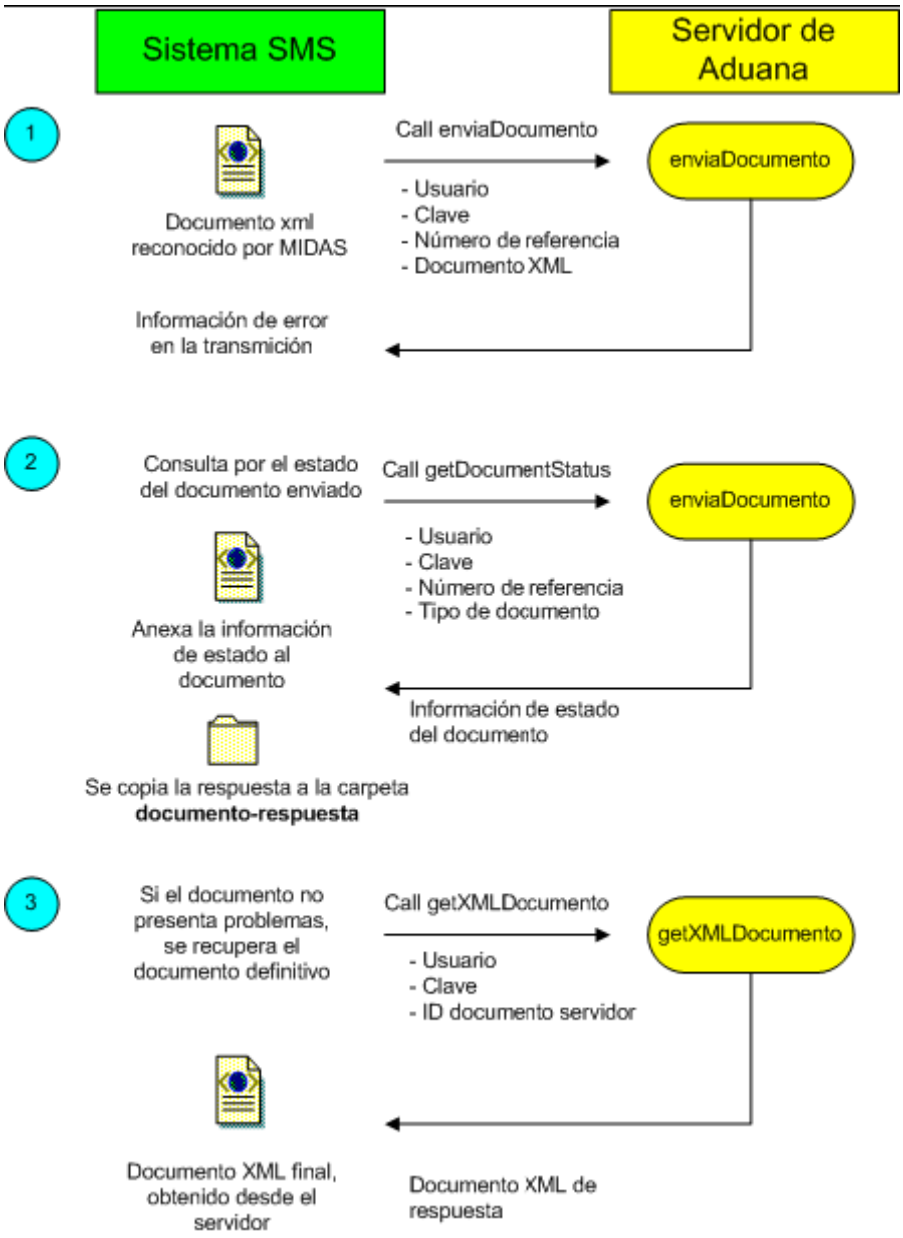

Figura 2: Esta figura muestra las llamadas desde SMS (izquierda) a los WebServices (derecha). En el diagrama se incluye los parámetros enviados, al igual que la información retornada por los WebServices y el manejo de las respuestas por parte de SMS.

Los servicios

Aduana posee un servicio para envío y otro para recepción de mensajería por SOAP. Dichos servicios son:

EnviaDocumento

Descripción:

Realiza el envió de documentos XML por SOAP.

Parámetros:

Login	: Usuario del sistema de seguridad SNA
Clave	: Clave del sistema de seguridad del SNA.
NumeroReferencia	: Número de referencia asignado por el emisor.
Xml	: String con el archivo xml que se desea enviar.

Uri:

Representa al objeto remoto, en este caso “Documentos”.

Método:

El método se llama “enviaDocumento”.

GetDocumentStatus

Descripción:

Realiza la consulta de las respuestas, disponibles para un envío efectuado con anterioridad.

Parámetros:

User	: Usuario del sistema de seguridad SNA
Pwd	: Clave del sistema de seguridad del SNA.
Referente	: Número de referencia asignado por el emisor.
DocType	: Tipo de documento (Ej: MIC).

Uri:

Representa al objeto remoto, en este caso “Documentos”.

Método:

El método se llama "getDocumentStatus".

getXMLDocumento

Descripción:

Entrega un documento XML definitivo, con los tags de respuestas asignados por el servidor de Aduanas. Este documento, reemplaza al documento original

Parámetros:

Login	: Usuario del sistema de seguridad SNA
Clave	: Clave del sistema de seguridad del SNA.
Id	: Id del documento entregado por el servidor, al llamar a **getDocumentStatus**

Uri:

Representa al objeto remoto, en este caso “Documentos”

Método:

El método se llama "getDocumentStatus".

Invocación de servicios

Para invocar los servicios web, se tienen que realizar los siguientes pasos:

1.- Definimos la url. Esta representa el punto final en el que reside el servicio

Ej: URL url = new URL(“http://200.72.133.20/soap/servlet/rpcrouter”);

2.- Creamos el objeto org.apache.soap.rpc.RPCMessage.call. Este objeto es la interfaz principal, para ejecutar los servicios SOAP.

Ej: Call call = new Call();

El objeto Callrepresenta la llamada a un procedimiento remoto, mediante el envío de un mensaje XML (SOAP).

3.- Definimos la URI objetivo dentro del objeto Call usando el método “setTargetObjectURI”

Ej: call.setTargetObjectURI("urn:tuxpan:Documentos");

Esto es utilizado para referenciar un objeto asociado a un descriptor de despliegue, en este caso el objeto Documentos.

4.-Definimos el método a invocar, dentro del objeto Call.

Ej: call.setMethodName(metodo)

Para nuestro ejemplo, la variable “metodo” podrá tomar los valores “enviaDocumento” o “getDocumentStatus”, ambos del objeto “Documentos”.

5.- Definimos la codificación

Ej: call.setEncodingStyleURI(Constants.NS_URI_SOAP_ENC)

6.- Asignamos los parámetros que serán enviados al servicio remoto
Ej:

```
Vector params = new Vector();
call.setParams(params);
if (nombreParametros != null) {
        for (int j=0; j<nombreParametros.length; j++) {
                params.addElement(new Parameter(nombreParametros[j], String.class,
                valorParametros[j], null));
        }
}
```

En el ejemplo, se crea un objeto **Vector** que contendrá los parámetros a enviar. Se define este objeto dentro del objeto **Call**, como portador de parámetros. Luego se llena el vector con objetos **Parameter** con nombre del parámetro, tipo de dato, valor del parámetro y null.

7.- Se invoca al servicio configurado con los pasos anteriores

Ej: Response response = call.invoke(url, "");

El ejemplo muestra la creación de un objeto **Response**, el cual contendrá la respuesta de la invocación. La invocación se realiza llamando al método **invoke** del objeto **Call** que antes habíamos creado.

8.- Chequeamos el objeto **Response** para ver si se ha generado algún fallo usando el método generatedFault().

```
Ej:     if( !response.generatedFault() ) {
        ....
        }
```

9.- Por último, si el servicio retorna alguna respuesta, recuperamos la respuesta.

```
Ej:     Parameter result = response.getReturnValue();
        if (result != null)
                return result.getValue();
        else
                return null;
```

Se define un objeto **Parameter** al cual le asignaremos la salida del método **getReturnValue** del objeto response. Luego recuperaremos el valor de la respuesta con el método **getValue** del objeto **Parameter** creado.

Los fragmentos de códigos mostrados, corresponden a la clase ServidorSOAP método invoca Servicio del sistema SMS.

Referencias

1.- Documentación de Apache SOAP
http://ws.apache.org/soap/docs/index.html

2.- Información sobre documentos de Aduana
http://www.aduana.cl
Entrar a la opción Isidoro → Módulos

3.- Web site de Java
http://java.sun.com