

DIVISION DE SISTEMAS DE INFORMACION Y COMUNICACIONES



Web Services de Port@firmas v3.2.6

Integración de aplicaciones con Port@firmas

Documento de Integración
Sistemas Desarrollo
Versión del documento
rev001
Fecha de revisión
30/07/2013
Realizado por
Sistemas Desarrollo

1 Control de modificaciones....4
1.1 Versión actual del documento....4
1.2 Versiones anteriores del documento....4
2 WebServices de Portafirmas....5
3 Servicio de consulta....5
3.1 downloadDocument....6
3.2 downloadSign....6
3.3 queryDocumentTypes....7
3.4 queryJobs....8
3.5 queryRequest....9
3.6 queryStates....10
3.7 queryUsers....11
3.8 queryImportanceLevels....12
3.9 getCVS....13
3.10 queryEnhancedUsers....13
3.11 queryEnhancedJobs....14
3.12 queryEnhancedUserJobAssociatedToJob....15
3.13 queryEnhancedUserJobAssociatedToUser....16
3.14 querySeats....16
4 Servicio de Modificación....17
4.1 createRequest....18
4.2 deleteDocument....18
4.3 deleteRequest....19
4.4 deleteSigners....20
4.5 insertDocument....21
4.6 insertSigners....21
4.7 sendRequest....22
4.8 updateRequest....23
4.9 removeRequest....24
5 Servicio de administración (AdminService)....25
5.1 insertEnhancedUsers....25
5.2 updateEnhancedUsers....26
5.3 deleteUsers....27
5.4 insertEnhancedJobs....28
5.5 updateEnhancedJobs....29
5.6 deleteJobs....30
5.7 assignJobToUser....30
5.8 separateJobToUser....32
6 Definición de elementos....34
6.1 Token de autenticación (authentication)....34
6.2 Lista de firmantes (signerList)....34
6.3 Usuarios y Cargos (User, Job y UserJobList)....36

6.4 Documentos (document y documentType)....37
6.5 Peticiones (request)....39
6.6 Tipo de firma (signType)....43
6.7 Lista de líneas de firma (signLineList)....44
6.8 Lista de remitentes (remitterList)....44
6.9 Lista de avisos (noticeList)....45
6.10 Lista de estados (stateList, state)....45
6.11 Firma (signature)....46
6.12 Nivel de importancia (importanceLevel)....47
6.13 Estado de la petición (requestStatus)....47
6.14 Lista de comentarios (commentList)....47
6.15 Información sello de tiempo (timestampInfo)....48
6.16 Parametros (parameter)....48
6.17 Lista de parámetros (parameterList)....49
6.18 Sede (seat)....49
6.19 Lista de sedes (seatList)....49
6.20 Información ampliada de usuario y cargo (enhancedUserJobInfo)....50
6.21 Lista de usuarios con información ampliada (enhancedUserList)....50
6.22 Lista de cargos con información ampliada (enhancedJobList)....51
6.23 Asociación entre usuario y cargo (enhancedUserJobAssociated)....52
6.24 Lista de asociaciones entre usuario y cargo (enhancedUserJobAssociatedList)....52
7 Servicio de respuesta para aplicaciones que se integran con Portafirmas....53

1 Control de modificaciones

1.1 Versión actual del documento

Revisión Actual: 001
Fecha: 30/07/2013
Autor: Sistemas Desarrollo
Descripción de los cambios:

- Ningún cambio con respecto a la anterior versión.

1.2 Versiones anteriores del documento

Documento: Guia_WebServices_Portafirmas_Portafirmas_v3.2.5_14_rev001.
Fecha: 26/03/2013
Autor: Sistemas Desarrollo

2 WebServices de Portafirmas

El portafirmas de versión 3 tiene una serie de webservices, a los que se ha llamado webservices de versión 2. Dichos webservices son:

- **Servicio de consulta:** Permite consultar información sobre las peticiones que hay en el servidor, así como información sobre tablas maestras
- **Servicio de modificación:** Permite crear y modificar peticiones
- **Servicio de administración**: Permite crear, modificar y eliminar usuarios y cargos de la aplicación. Las aplicaciones que deseen consumir este servicio deberán tener unos permisos especiales en la aplicación.

3 Servicio de consulta

El wsdl del servicio se encuentra en

http://servidor/portafirma/servicesv2/QueryService?wsdl

El servicio de consulta permite realizar consultas sobre peticiones existentes en el portafirmas, así como recuperar datos de tablas maestras. Las funcionalidades disponibles son las siguientes:

- **downloadDocument:** Permite descargar documentos anexos a una solicitud
- **downloadSign:** Permite descargar la firma de un documento
- **queryDocumentTypes:** Permite recuperar una lista con los tipos de documentos disponibles
- **queryJobs:** Permite recuperar una lista con los cargos disponibles en la aplicación
- **queryRequest:** Permite recuperar el detalle de una petición existente en el portafirmas
- **queryStates:** Permite recuperar una lista con los estados en los que puede estar una petición
- **queryUsers:** Permite recuperar una lista de usuarios
- **queryImportanceLevels**: Permite recuperar una lista de los niveles de importancia
- **queryEnhancedUsers**: Permite recuperar una lista de usuarios con información ampliada.
- **queryEnhancedJobs**: Permite recuperar una lista de cargos con información ampliada.
- **queryEnhancedUserJobAssociatedToJob**: Permite recuperar información ampliada sobre los usuarios asociados a un cargo.
- **queryEnhancedUserJobAssociatedToUser:** Permite recuperar la información ampliada de los cargos asociados a un usuario.
- **querySeats**: Permite recuperar las sedes existentes en la aplicación.

3.1 downloadDocument

El método downloadDocument permite recuperar el contenido de un documento a partir del id del mismo. La petición responde al siguiente esquema:

```
<xsd:element name="downloadDocument">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication"/>
			<xsd:element name="documentId" type="xsd:string"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 6.1
- **documentId:** Id del documento

Y la respuesta:

```
<xsd:element name="downloadDocumentResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="documentBinary" type="xsd:base64Binary"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **documentBinary:** Contenido del fichero en base 64

3.2 downloadSign

El método downloadSign permite recuperar el contenido de la firma de un documento a partir del id del documento. La petición responde al siguiente esquema:

```
<xsd:element name="downloadSign">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication"/>
			<xsd:element name="documentId" type="xsd:string"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **documentId:** Id del documento

Y la respuesta:

```
<xsd:element name="downloadSignResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="signature" type="type:signature"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>

<xsd:complexType name="signature">
	<xsd:sequence>
		<xsd:element minOccurs="0" name="identifier" nillable="true"
	type="xsd:string"/>
		<xsd:element minOccurs="0" name="sign" nillable="true"
	type="xsd:boolean"/>
		<xsd:element minOccurs="0" name="signFormat" nillable="true"
	type="type:signFormat"/>
		<xsd:element minOccurs="0" name="content" nillable="true"
	type="xsd:base64Binary" xmime:expectedContentTypes="application/octet-
	stream"/>
	</xsd:sequence>
</xsd:complexType>
```

- **identifier:** Identificador del documento
- **sign:** Indica si el documento está o no firmado
- **signFormat:** Formato de la firma (PKCS7, CMS, CADES, XADES, XADES IMPLICITO, XADES EXPLICITO, XADES ENVELOPING, PDF)
- **content:** Contenido en base 64 de la firma

3.3 queryDocumentTypes

El método queryDocumentTypes permite recuperar los tipos de documento disponibles:

```
<xsd:element name="queryDocumentTypes">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication"/>
			<xsd:element name="query" type="xsd:string"/>
		</xsd:sequence>
	</xsd:complexType>
```

```
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **query:** Cadena con el código del tipo de documento a buscar

Y la respuesta:

```
<xsd:element name="queryDocumentTypesResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="documentTypeList"
		type="type:documentTypeList"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>

<xsd:complexType name="documentTypeList">
	<xsd:sequence>
		<xsd:element maxOccurs="unbounded" minOccurs="0"
	name="documentType" type="type:documentType"/>
	</xsd:sequence>
</xsd:complexType>
```

- **documentTypeList:** Lista con los tipos de documento recuperados. Contiene uno o varios elementos de tipo documentType, definidos en el punto 6.4

3.4 queryJobs

El método queryJobs permite recuperar los cargos disponibles:

```
<xsd:element name="queryJobs">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication"/>
			<xsd:element name="query" type="xsd:string"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **query:** Cadena con el código cargo a buscar

Y la respuesta:

```
<xsd:element name="queryJobsResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="jobList" type="type:jobList"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>

<xsd:complexType name="jobList">
	<xsd:sequence>
		<xsd:element maxOccurs="unbounded" minOccurs="0" name="job"
	type="type:job"/>
	</xsd:sequence>
</xsd:complexType>

<xsd:complexType name="job">
	<xsd:complexContent>
		<xsd:extension base="type:userJob">
			<xsd:sequence>
				<xsd:element minOccurs="0" name="description" nillable="true"
			type="xsd:string"/>
			</xsd:sequence>
		</xsd:extension>
	</xsd:complexContent>
</xsd:complexType>

<xsd:complexType abstract="true" name="userJob">
	<xsd:sequence>
		<xsd:element minOccurs="0" name="identifier" nillable="true"
	type="xsd:string"/>
	</xsd:sequence>
</xsd:complexType>
```

- **identifier:** Identificador del cargo
- **description:** Descripción del cargo

3.5 queryRequest

El método queryRequest permite recuperar una petición existente en el servidor. Los usuarios sólo podrán acceder a las peticiones creadas por ellos mismos:

```
<xsd:element name="queryRequest">
	<xsd:complexType>
		<xsd:sequence>
```

```
                <xsd:element name="authentication" type="type:authentication"/>
                <xsd:element name="requestId" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **requestId:** Identificador de la petición

Y la respuesta:

```
<xsd:element name="queryRequestResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="request" type="type:request"/>
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **request:** Elemento complejo que representa una petición. Se define en el punto 6.5

3.6 queryStates

El método queryDocumentTypes permite recuperar los cargos disponibles:

```
<xsd:element name="queryStates">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="authentication" type="type:authentication"/>
                <xsd:element name="query" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **query:** Cadena con el código del estado a buscar

Y la respuesta:

```
<xsd:element name="queryStatesResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="stateList" type="type:stateList"/>
```

```
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **stateList:** Lista de estados recuperada. El elemento stateList se define en el punto 6.10

3.7 queryUsers

El método queryUsers permite recuperar los usuarios disponibles:

```
<xsd:element name="queryUsers">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="authentication" type="type:authentication"/>
                <xsd:element name="query" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **query:** Cadena con el identificador del usuario a buscar (Habitualmente su DNI)

Y la respuesta:

```
<xsd:element name="queryUsersResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="userList" type="type:userList"/>
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>



<xsd:complexType name="userList">
        <xsd:sequence>
            <xsd:element maxOccurs="unbounded" minOccurs="0" name="user"
        type="type:user"/>
        </xsd:sequence>
</xsd:complexType>
```

- **identifier:** Identificador del usuario (Habilitualmente el DNI)

- **userList:** Respresenta una lista de usuarios. La lista tiene varios subelementos de tipo user. El tipo user viene definido en el punto 6.3

3.8 queryImportanceLevels

El método queryImportanceLevels permite recuperar los niveles de importancia disponibles:

```
<xsd:element name="queryImportanceLevels">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication"/>
			<xsd:element name="query" type="xsd:string"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **query:** Cadena con el código del nivel de importancia a buscar

Y la respuesta:

```
<xsd:element name="queryImportanceLevelsResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="importanceLevelList"
			type="type:importanceLevelList"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>


<xsd:complexType name="importanceLevelList">
	<xsd:sequence>
		<xsd:element maxOccurs="unbounded" minOccurs="0" name="user"
	type="type:importanceLevel"/>
	</xsd:sequence>
</xsd:complexType>
```

- **importanceLevelList:** Listado de niveles de importancia

3.9 getCVS

El método getCVS permite generar el Código de Verificación Seguro de una firma de un documento:

```
<xsd:element name="getCVS">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication"/>
			<xsd:element name="firma" type="type:signature"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **firma:** Firma electrónica de un documento. Ver el punto 4.11

Y la respuesta:

```
<xsd:element name="getCVSResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="cvs" type="xsd:string"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **cvs:** Cadena de texto con el Código de Verificación Seguro.

3.10 queryEnhancedUsers

Permite recuperar una lista de usuarios válidos con algo más de información a partir de una consulta. El tipo asociado a la petición es el siguiente:

```
<xsd:element name="queryEnhancedUsers">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication" />
			<xsd:element name="queryUser" type="xsd:string" nillable="true" />
			<xsd:element name="querySeat" type="xsd:string" nillable="true"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 6.1
- **queryUser:** Puede contener o bien el identificador del usuario o bien el nombre y los apellidos del usuario que se desea buscar. Si es nulo no se aplicará el filtro.
- **querySeat**: Contiene el código de la sede de la que se desean recuperar los usuarios. Si es nulo no se aplicará el filtro.

La respuesta:

```
<xsd:element name="queryEnhancedUsersResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="enhancedUserList"
type="type:enhancedUserList" />
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **enhancedUserList**: Lista de usuarios que coinciden con la consulta. Ver el punto 6.21

3.11 queryEnhancedJobs

Permite recuperar una lista de cargos válidos con algo más de información a partir de una consulta. El tipo asociado a la petición es el siguiente:

```
<xsd:element name="queryEnhancedJobs">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication" />
			<xsd:element name="queryJob" type="xsd:string" nillable="true" />
			<xsd:element name="querySeat" type="xsd:string" nillable="true"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 6.1
- **queryUser:** Puede contener o bien el identificador del cargo o bien el nombre completo del cargo. Si es nulo no se aplicará el filtro.
- **querySeat**: Contiene el código de la sede de la que se desean recuperar los cargos. Si es nulo no se aplicará el filtro.

La respuesta:

```
<xsd:element name="queryEnhancedJobsResponse">
	<xsd:complexType>
```

```
            <xsd:sequence>
                <xsd:element name="enhancedJobList"
type="type:enhancedJobList" />
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **enhancedJobList**: Lista de cargos que coinciden con la consulta. Ver el punto 6.22

3.12 queryEnhancedUserJobAssociatedToJob

Recupera la lista de usuarios asociados a un cargo. Un cargo sólo puede tener asociado a un usuario en un determinado periodo de tiempo.

```
<xsd:element name="queryEnhancedUserJobAssociatedToJob">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="authentication" type="type:authentication" />
                <xsd:element name="jobIdentifier" type="xsd:string"/>
                </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 6.1
- **jobIdentifier:** Identificador del cargo del que se quieren saber los usuarios asociados.

La respuesta:

```
<xsd:element name="queryEnhancedUserJobAssociatedToJobResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="enhancedUserJobAssociatedList"
type="type:enhancedUserJobAssociatedList"/>
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **enhancedUserJobAssociatedList:** Aparecerá un elemento **enhancedUserJobAssociated** por cada asociación cargo-usuario que exista. Ver el punto 6.24

Errores:

– El cargo no existe o no está vigente.

3.13 queryEnhancedUserJobAssociatedToUser

Recupera la lista de cargos asociados a un usuario. Un usuario sólo puede tener asociado un cargo en un determinado periodo de tiempo.

```
<xsd:element name="queryEnhancedUserJobAssociatedToUser">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication" />
			<xsd:element name="userdentifier" type="xsd:string"/>
			</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 6.1
- **userdentifier:** Identificador del usuario del que se quieren saber los cargos asociados.

La respuesta:

```
<xsd:element name="queryEnhancedUserJobAssociatedToUserResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="enhancedUserJobAssociatedList"
type="type:enhancedUserJobAssociatedList"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **enhancedUserJobAssociatedList:** Aparecerá un elemento **enhancedUserJobAssociated** por cada asociación cargo-usuario que exista. Ver el punto 6.24

Errores:

– El usuario no existe o no está vigente.

3.14 querySeats

Recupera las sedes que cumplen una determinada condición.

```
<xsd:element name="querySeats">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication" />
			<xsd:element name="query" type="xsd:string" nillable="true"/>
				</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 6.1
- **query:** Código o nombre de la sede. Si es nulo se devolverán todas.

Respuesta:

```
<xsd:element name="querySeatsResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="seatList" type="type:seatList" />
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **seatList:** Lista de sedes que cumplen con la condición de la consulta. Ver punto 6.19

4 Servicio de Modificación

El wsdl del servicio se encuentra en

http://servidor/portafirma/servicesv2/ ModifyService?wsdl

El servicio de modificación permite crear y modificar peticiones existentes en el portafirmas. Las funcionalidades disponibles son las siguientes:

- **createRequest:** Permite crear en el servidor de portafirmas una nueva petición. Dicha petición no será enviada hasta que se invoque la función sendRequest.
- **deleteDocument:** Permite quitar un documento de una petición existente
- **deleteRequest:** Permite borrar una petición previamente creada.
- **deleteSigners:** Permite borrar una lista de firmantes de una o varias líneas de firma
- **insertDocument:** Permite añadir un documento a una petición ya existente
- **insertSigners:** Permite insertar un conjunto de firmantes en una línea de firma

- **sendRequest:** Envía una petición previamente creada
- **updateRequest**: Permite actualizar una petición previamente creada
- **removeRequest**: Permite retirar una petición que aún no está finalizada.

4.1 createRequest

El método createRequest sirve para crear una petición en el portafirmas. Para que una petición llegue al buzón de los destinatarios es necesario crearla y después enviarla. Se recomienda crear la petición sin documentos y añadirlos mediante el método insertDocument. La ventaja de este método es que al llamar a insertDcument, obtendremos directamente el id del documento sin tener que llamar a queryRequest.

```
<xsd:element name="createRequest">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication"/>
			<xsd:element name="request" type="type:request"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **request:** Elemento complejo que representa una petición. Se define en el punto 6.5

Y la respuesta:

```
<xsd:element name="createRequestResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="requestId" type="xsd:string"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **identifier: requestId:** Identificador de la petición

4.2 deleteDocument

El método deleteDocument permite quitar un documento de una petición que ha sido creada, pero que todavía no ha sido enviada:

```
<xsd:element name="deleteDocument">
	<xsd:complexType>
```

```
            <xsd:sequence>
                <xsd:element name="authentication" type="type:authentication"/>
                <xsd:element name="documentId" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **documentId:** Identificador del documento

Y la respuesta:

```
<xsd:element name="deleteDocumentResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="documentId" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **documentId:** Identificador del documento

4.3 deleteRequest

El método deleteRequest permite eliminar una petición que ha sido creada, pero que todavía no ha sido enviada:

```
<xsd:element name="deleteRequest">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="authentication" type="type:authentication"/>
                <xsd:element name="requestId" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **requestId:** Identificador de la petición

Y la respuesta:

```
<xsd:element name="deleteRequestResponse">
        <xsd:complexType>
            <xsd:sequence>
```

```
                <xsd:element name="requestId" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **requestId:** Identificador de la petición

4.4 deleteSigners

El método deleteSigners permite eliminar firmantes de una petición que ha sido creada, pero que todavía no ha sido enviada:

```
<xsd:element name="deleteSigners">
        <xsd:complexType>
                <xsd:sequence>
                        <xsd:element name="authentication" type="type:authentication"/>
                        <xsd:element name="requestId" type="xsd:string"/>
                        <xsd:element minOccurs="0" name="signLineNumber" nillable="true"
                type="xsd:int"/>
                        <xsd:element name="signerList" type="type:signerList"/>
                </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **requestId:** Identificador de la petición
- **signLineNumber:** Número de linea del que se quiere borrar al destinatario. La primera linea viene representada por un 0. Si el número de linea está vacío, los elementos de la linea de firmantes “signerList”, se borran de todas las lineas en las que estén presentes
- **signerList:** Lista de cargos y usuarios que se quieren quitar de la linea de firma. Se define en el punto 6.2

Y la respuesta:

```
<xsd:element name="deleteSignersResponse">
        <xsd:complexType>
                <xsd:sequence>
                        <xsd:element name="requestId" type="xsd:string"/>
                </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **requestId:** Identificador de la petición

4.5 insertDocument

El método insertDocument permite añadir un documento a una petición que ha sido creada, pero que todavía no ha sido enviada:

```
<xsd:element name="insertDocument">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication"/>
			<xsd:element name="requestId" type="xsd:string"/>
			<xsd:element name="document" type="type:document"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **requestId:** Identificador de la petición
- **document:** Elemento complejo que define un documento y su contenido. Se define en el punto 6.4

Y la respuesta:

```
<xsd:element name="insertDocumentResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="documentId" type="xsd:string"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **documentId:** Identificador del nuevo documento

4.6 insertSigners

El método insertSigners permite añadir una nueva linea de firma a una petición que ha sido creada, pero que todavía no ha sido enviada:

```
<xsd:element name="insertSigners">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication"/>
```

```
                <xsd:element name="requestId" type="xsd:string"/>
                <xsd:element name="signLine" type="xsd:int"/>
                <xsd:element name="signerList" type="type:signerList"/>
                <xsd:element name="signLineType" type="type:signLineType"/>
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **requestId:** Identificador de la petición
- **signLine:** Número de linea en la que se quiere insertar la linea nueva. Si el número de linea es 0, se insertará en el primer lugar. Si el número de linea es igual al número de lineas que ya existe, la linea se insertará la última.
- **signerList:** Lista de cargos y usuarios que se quieren quitar de la linea de firma. Se define en el punto 6.2
- **signLineType:** Indica si una linea de firma debe ser firmada, o se debe dar el visto bueno. Los valores posibles son FIRMA y VISTOBUENO.

Y la respuesta:

```
<xsd:element name="insertSignersResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="requestId" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **requestId:** Identificador de la petición

4.7 sendRequest

El método sendRequest permite enviar una petición que ha sido creada, pero que todavía no ha sido enviada. Dicha petición aparecerá en la bandeja de entrada de los usuarios correspondientes.

```
<xsd:element name="sendRequest">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="authentication" type="type:authentication"/>
                <xsd:element name="requestId" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **requestId:** Identificador de la petición

Y la respuesta:

```
<xsd:element name="sendRequestResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="requestId" type="xsd:string"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **requestId:** Identificador de la petición

4.8 updateRequest

El método updateRequest permite actualizar la información una petición que ha sido creada, pero que todavía no ha sido enviada.

```
<xsd:element name="updateRequest">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication"/>
			<xsd:element name="request" type="type:request"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **request:** Elemento complejo que representa una petición. Se define en el punto 6.5

Y la respuesta:

```
<xsd:element name="updateRequestResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="requestId" type="xsd:string"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **requestId:** Identificador de la petición

4.9 removeRequest

El método removeRequest permite retirar una petición enviada que no se encuentra en un estado final. Solo puede retirar una petición el usuario que la creó.

```
<xsd:element name="removeRequest">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication" />
			<xsd:element name="requestId" type="xsd:string" />
			<xsd:element name="removingMessage" type="xsd:string" />
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 4.1
- **requestId:** Identificador de la petición.
- **removingMessage:** Mensaje de usuario con el motivo del retiro de la petición.

Y la respuesta:

```
<xsd:element name="removeRequestResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="requestId" type="xsd:string"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **requestId:** Identificador de la petición retirada.

5 Servicio de administración (AdminService)

El wsdl del servicio se encuentra en

http://servidor/portafirma/servicesv2/AdminService?wsdl

El servicio de administración permite gestionar usuarios de la aplicación. Las aplicaciones consumidoras de este servicio deberán tener permisos especiales en Port@firmas, más concretamente deberán especificar para qué sedes desean tener permisos de administración.

Las funcionalidades disponibles son las siguientes:

- **insertEnhancedUsers:** Da de alta usuarios en la aplicación.
- **updateEnhancedUsers:** Modifica usuarios ya existentes en la aplicación
- **deleteUsers:** Elimina usuarios de la aplicación.
- **insertEnhancedJobs:** Inserta cargos en la aplicación.
- **updateEnhancedJobs:** Modifica cargos en la aplicación.
- **deleteJobs:** Elimina cargos en la aplicación.
- **assignJobToUser:** Asigna un cargo a un usuario para un periodo de tiempo.
- **separateJobToUser:** Desasigna un cargo de un usuario.

5.1 insertEnhancedUsers

Crea nuevos usuarios en la aplicación. Si alguno de los usuarios existiera y estuviera vigente dará error. En caso de existir y no estar vigente se modificará para que lo estuviera.

Petición:

```
<xsd:element name="insertEnhancedUsers">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication" />
			<xsd:element name="enhancedUserList"
type=type:enhancedUserList" />
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 6.1
- **enhancedUserList:** Lista de usuarios a insertar

Respuesta:

```
<xsd:element name="insertEnhancedUsersResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="enhancedUsersInserted" type="xsd:integer" />

		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **enhancedUsersInserted:** Número de usuarios insertados.

Errores:

- La sede no existe.
- La aplicación no tiene permisos para crear usuarios en la sede del nuevo usuario.
- Existe un usuario vigente con el mismo identificador.

5.2 updateEnhancedUsers

Modifica usuarios en la aplicación. Si alguno de los usuarios no existiera o no estuviera vigente dará error.

Petición:

```
<xsd:element name="updateEnhancedUsers">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication" />
			<xsd:element name="enhancedUserList"
type="type:enhancedUserList" />
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 6.1
- **enhancedUserList:** Lista de usuarios a modificar.

Respuesta:

```
<xsd:element name="updateEnhancedUsersResponse">
```

```
        <xsd:complexType>
                <xsd:sequence>
                        <xsd:element name="enhancedUsersUpdated" type="xsd:integer" />

                </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **enhancedUsersUpdated:** Número de usuarios modificados.

Errores:

- La sede no existe.
- La aplicación no tiene permisos para modificar usuarios en la sede.
- El usuario no existe o no está vigente.

5.3 deleteUsers

Elimina usuarios de la aplicación. Si algún usuario no existe dará error. Si el usuario tiene relaciones que no pueden eliminarse (peticiones, autorizaciones sobre otros usuarios...) el usuario se modificará para que quede como no vigente.

Petición:

```
<xsd:element name="deleteUsers">
        <xsd:complexType>
                <xsd:sequence>
                        <xsd:element name="authentication" type="type:authentication" />
                        <xsd:element name="userIdentifierList" type="type:stringList" />
                </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 6.1
- **userIdentifierList:** Lista de identificadores de usuarios a eliminar.

Respuesta:

```
<xsd:element name="deleteUsersResponse">
        <xsd:complexType>
                <xsd:sequence>
                        <xsd:element name="usersDeleted" type="xsd:integer" />
                </xsd:sequence>
        </xsd:complexType>
```

```
</xsd:element>
```

- **usersDeleted:** Número de usuarios borrados.

Errores:

- El usuario no existe.
- La aplicación no tiene permisos sobre la sede del usuario.

5.4 insertEnhancedJobs

Crea nuevos cargos en la aplicación. Si alguno de los cargos existiera y estuviera vigente daría error. En caso de existir y no estar vigente se modificaría para que lo estuviera.

Petición:

```
<xsd:element name="insertEnhancedJobs">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication" />
			<xsd:element name="enhancedJobList"
type=type:enhancedUserList" />
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 6.1
- **enhancedJobList:** Lista de usuarios a insertar

Respuesta:

```
<xsd:element name="insertEnhancedJobsResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="enhancedJobsInserted" type="xsd:integer" />

		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **enhancedUsersInserted:** Número de cargos insertados.

Errores:

- La sede no existe.
- La aplicación no tiene permisos para crear cargos en la sede del nuevo cargo.
- Existe un cargo vigente con el mismo identificador.
- Existe un cargo con la misma descripción en la misma sede.

5.5 updateEnhancedJobs

Modifica cargos en la aplicación. Si alguno de los cargos no existiera o no estuviera vigente dará error.

Petición:

```
<xsd:element name="updateEnhancedJobs">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication" />
			<xsd:element name="enhancedJobList"
type="type:enhancedJobList" />
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 6.1
- **enhancedJobList:** Lista de usuarios a modificar.

Respuesta:

```
<xsd:element name="updateEnhancedJobsResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="enhancedJobsUpdated" type="xsd:integer" />

		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **enhancedJobsUpdated:** Número de cargos modificados.

Errores:

- La sede no existe.
- La aplicación no tiene permisos para modificar cargos en la sede.

- El cargo no existe o no está vigente.

5.6 deleteJobs

Elimina cargos de la aplicación. Si algún cargo no existe dará error. Si el cargo tiene relaciones que no pueden eliminarse (usuarios asociados, peticiones) el cargo se modificará para que quede como no vigente.

Petición:

```
<xsd:element name="deleteJobs">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication" />
			<xsd:element name="jobIdentifierList" type="type:stringList" />
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 6.1
- **jobIdentifierList:** Lista de identificadores de cargos a eliminar.

Respuesta:

```
<xsd:element name="deleteJobsResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="jobsDeleted" type="xsd:integer" />
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **jobsDeleted:** Número de cargos borrados.

Errores:

- El cargo no existe.
- La aplicación no tiene permisos sobre la sede del cargo.

5.7 assignJobToUser

Asigna un cargo a un usuario para un determinado periodo de tiempo. Si ya existiera la relación usuario-cargo, se modificaría el periodo de vigencia.

Petición:

```
<xsd:element name="assignJobToUser">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication" />
			<xsd:element name="jobIdentifier" type="xsd:string" />
			<xsd:element name="userIdentifier" type="xsd:string" />
			<xsd:element name="fstart" type="xsd:date" nillable="true"
minOccurs="0"/>
			<xsd:element name="fend" type="xsd:date" nillable="true"
minOccurs="0"/>
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 6.1
- **jobIdentifier:** Identificador del cargo que se quiere asignar al usuario.
- **userIdentifier:** Identificador del usuario al que se le quiere asignar el cargo.
- **fstart**: Fecha de inicio de la asociación usuario-cargo. Si es nula se considerará el momento actual.
- **fend**: Fecha de finalización de la asociación usuario-cargo. Si es nula se considerará que el periodo de vigencia de asociación no tiene fin.

Respuesta:

```
<xsd:element name="assignJobToUserResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="assigned" type="xsd:boolean" />
		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **assigned**: Indica si se ha podido hacer la asociación.

Errores:

- Fecha de fin anterior a fecha de inicio.
- El cargo no existe o no está vigente
- El usuario no existe o no está vigente.
- La aplicación no tiene permisos sobre la sede del usuario.
- La aplicación no tiene permisos sobre la sede del cargo.
- El cargo está asociado a otro usuario en ese periodo (o en un periodo que se solapa).

- El usuario está asociado a otro cargo en ese periodo (o en un periodo que se solapa).

5.8 separateJobToUser

Elimina la relación usuario-cargo.

Petición:

```
<xsd:element name="separateJobToUser">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="authentication" type="type:authentication" />
			<xsd:element name="jobIdentifier" type="xsd:string" />
			<xsd:element name="userIdentifier" type="xsd:string" />

		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **authentication:** Token de autenticación. Ver el punto 6.1
- **jobIdentifier:** Identificador del cargo del que se quiere eliminar la asociación.
- **userIdentifier:** Identificador del usuario del que se quiere eliminar la asociación.

Respuesta:

```
<xsd:element name="separateJobToUserResponse">
	<xsd:complexType>
		<xsd:sequence>
			<xsd:element name="separated" type="xsd:boolean" />

		</xsd:sequence>
	</xsd:complexType>
</xsd:element>
```

- **separated**: Indica si se ha eliminado la asociación.

Errores:

- El cargo no existe.
- El usuario no existe.
- La aplicación no tiene permisos sobre la sede del usuario.
- La aplicación no tiene permisos sobre la sede del cargo.

– El cargo no está asociado al usuario.

6 Definición de elementos

6.1 Token de autenticación (authentication)

Todos los mensajes SOAP que se envían hacia el servidor tienen un token de autenticación que es igual en todos los casos. El token de autenticación responde al siguiente esquema:

```
<xsd:complexType name="authentication">
	<xsd:sequence>
		<xsd:element name="userName" nillable="false" type="xsd:string"/>
		<xsd:element name="password" nillable="false" type="xsd:string"/>
	</xsd:sequence>
</xsd:complexType>
```

- **userName:** Identificador del usuario. Se corresponde con la aplicación que quiere enviar el mensaje.
- **password:** Contraseña

La autenticación limita no sólo el acceso a la aplicación, si no que también limita el acceso a la información de las peticiones, así como su modificación. Un usuario sólo podrá ver y modificar aquellas peticiones que el mismo ha generado. Está limitación también se aplica a los documentos anexos a las peticiones así como a sus firmas.

Ejemplo:

```
<authentication>
	<userName>12345678Z</userName>
	<password>cambiame</password>
</authentication>
```

6.2 Lista de firmantes (signerList)

El elemento de tipo signerList, representa una lista de usuarios y cargos que deben firmar o dar el visto bueno a una petición.

```
<xsd:complexType name="signerList">
	<xsd:sequence>
		<xsd:element maxOccurs="unbounded" minOccurs="0" name="signer"
	type="type:signer"/>
	</xsd:sequence>
</xsd:complexType>
```

El elemento signerList, está formado por una lista de elementos del tipo signer que se define a continuación.

```
<xsd:complexType name="signer">
	<xsd:sequence>
		<xsd:element minOccurs="0" name="userJob" nillable="true"
	type="type:userJob"/>
		<xsd:element minOccurs="0" name="state" nillable="true"
	type="type:state"/>
		<xsd:element minOccurs="0" name="fstate" nillable="true"
	type="xsd:dateTime"/>
	</xsd:sequence>
</xsd:complexType>
```

- **userJob:** userJob es un elemento que puede representar tanto a un usuario de la aplicación como a un cargo. Se define en el punto 6.3
- **state:** Indica el estado de la linea de firma a la que pertenece el firmante. Este campo sólo está relleno cuando se recibe información desde el servidor y será ignorado si se envía para mdoficiar una petición. El tipo state viene definido en el punto 6.10
- **fstate:** Indica la fecha del último cambio de estado. Este campo sólo está relleno cuando se recibe información desde el servidor de portafirma, y será ignorado si se envía para mdoficiar una petición

Ejemplo de signer de tipo user

```
<signer>
	<user>
		<identifier>12345678Z</identifier>
		<name>Nombre</name>
		<surname1>Apellido1</surname1>
		<surname2>Apellido2</surname2>
	</user>
	<state>FIRMADO</state>
	<fstate>2010-10-26T21:32:52</fstate>
</signer>

<signer>
	<job>
		<identifier>DIRECTOR</identifier>
		<description>Director de la sección</description>
	</job>
	<state>FIRMADO</state>
	<fstate>2010-10-26T21:32:52</fstate>
</signer>
```

6.3 Usuarios y Cargos (User, Job y UserJobList)

El portafirmas tiene dos tipos de destinatarios a los que se pueden mandar las peticiones:

- Usuarios: Son personas o usuarios concretos. Vienen representados por el tipo User
- Cargos: Definen cargos que pueden ocupar los usuarios. Cada usuario puede ocupar un único cargo en un momento determinado. Cuando se envía una petición a un cargo, esta es recibida por todos los usuarios que ocupan dicho cargo.

Ambos tipos (user y job), son una extensión del tipo userJob. El tipo userJob se utilizará cuando sea necesario enviar listas con elementos de ambos tipos:

```
<xsd:complexType abstract="true" name="userJob">
	<xsd:sequence>
		<xsd:element minOccurs="0" name="identifier" nillable="true"
	type="xsd:string"/>
	</xsd:sequence>
</xsd:complexType>
```

- **identifier:** Identificador del usuario (Habilitualmente el DNI) o del cargo.

El tipo user, tiene algunos campos adicionales que identifican al usuario

```
<xsd:complexType name="user">
	<xsd:complexContent>
		<xsd:extension base="type:userJob">
			<xsd:sequence>
				<xsd:element minOccurs="0" name="name" nillable="true"
			type="xsd:string"/>
				<xsd:element minOccurs="0" name="surname1" nillable="true"
			type="xsd:string"/>
				<xsd:element minOccurs="0" name="surname2" nillable="true"
			type="xsd:string"/>
			</xsd:sequence>
		</xsd:extension>
	</xsd:complexContent>
</xsd:complexType>
```

- **name:** Nombre
- **surname1:** Primer Apellido
- **surname2:** Segundo Apellido

El tipo job, tiene un campo adicional con la descripción.

```
<xsd:complexType name="job">
	<xsd:complexContent>
		<xsd:extension base="type:userJob">
			<xsd:sequence>
				<xsd:element minOccurs="0" name="description" nillable="true"
			type="xsd:string"/>
			</xsd:sequence>
		</xsd:extension>
	</xsd:complexContent>
</xsd:complexType>
```

- **description:** Descripción del cargo

Ejemplos de user y job:

```
<user>
	<identifier>12345678Z</identifier>
	<name>Nombre</name>
	<surname1>Apellido1</surname1>
	<surname2>Apellido2</surname2>
</user>

<job>
	<identifier>DIRECTOR</identifier>
	<description>Director de la sección</description>
</job>
```

6.4 Documentos (document y documentType)

El tipo documento se utiliza para enviar al servidor información de un documento así como su contenido. También aparece dentro de las peticiones cuando se llama al método queryRequest, aunque en este caso el contenido está vacío.

```
<xsd:complexType name="document">
	<xsd:sequence>
		<xsd:element minOccurs="0" name="identifier" nillable="true"
	type="xsd:string"/>
		<xsd:element minOccurs="0" name="name" nillable="true"
	type="xsd:string"/>
		<xsd:element minOccurs="0" name="mime" nillable="true"
	type="xsd:string"/>
```

```
        <xsd:element minOccurs="0" name="documentType" nillable="true"
    type="type:documentType"/>
        <xsd:element minOccurs="0" name="content" nillable="true"
    type="xsd:base64Binary" xmime:expectedContentTypes="application/octet-
    stream"/>
        <xsd:element minOccurs="0" name="type" nillable="true"
    type="xsd:string"/>
        <xsd:element minOccurs="0" name="uri" nillable="true" type="xsd:string"/>
        <xsd:element minOccurs="0" name="sign" nillable="true"
    type="xsd:boolean"/>
    </xsd:sequence>
</xsd:complexType>
```

- **identifier:** Identificador del documento. Lo genera el portafirmas
- **name:** Nombre del documento (nombre del fichero)
- **mime:** Tipo mime del documento
- **documentType:** Tipo de documento. Los tipos de documentos se definen en la aplicación, e indican que tipo de contenido tiene el documento: Informe, petición, convenio, etc. La clasificación se hace de cara al usuario que lee el documento, pero no tiene ningún efecto en el comportamiento de la aplicación. El tipo de documento se define a continuación
- **type:** Campo no utilizado
- **uri:** Campo no utilizado
- **sign:** Indica si el documento debe ser firmado en las lineas de tipo firma (NO PROBADO)

```
<xsd:complexType name="documentType">
    <xsd:sequence>
        <xsd:element minOccurs="0" name="identifier" nillable="true"
    type="xsd:string"/>
        <xsd:element minOccurs="0" name="description" nillable="true"
    type="xsd:string"/>
        <xsd:element minOccurs="0" name="valid" nillable="true"
    type="xsd:boolean"/>
    </xsd:sequence>
</xsd:complexType>
```

- **identifier:** Identificador del tipo de documento
- **description:** Descripción
- **valid:** Indica si actualmente es válido, y se puede usar como tipo para un nuevo documento.

Ejemplo de elemento de tipo document:

```
<document>
    <identifier>12345678</identifier>
    <name>documento de prueba.pdf</name>
```

```
        <mime>application/pdf</mime>
        <documentType>
                <identifier>INFORME</identifier>
                <description>Informe genérico</description>
                <valid>true</valid>
        </documentType>
        <content>[[CONTENIDO EN BASE64]]</content>
        <type/>
        <uri/>
        <sign>true</sign>
</document>
```

Ejemplo de documentType:

```
<documentType>
        <identifier>INFORME</identifier>
        <description>Informe genérico</description>
        <valid>true</valid>
</documentType>
```

6.5 Peticiones (request)

El elemento de tipo request, representa las peticiones que se mandan a los usuarios para su firma. Una petición se compone de los siguientes elementos:

```
<xsd:complexType name="request">
        <xsd:sequence>
                <xsd:element minOccurs="0" name="identifier" nillable="true"
        type="xsd:string"/>
                <xsd:element minOccurs="0" name="subject" nillable="true"
        type="xsd:string"/>
                <xsd:element minOccurs="0" name="fentry" nillable="true"
        type="xsd:dateTime"/>
                <xsd:element minOccurs="0" name="fstart" nillable="true"
        type="xsd:dateTime"/>
                <xsd:element minOccurs="0" name="fexpiration" nillable="true"
        type="xsd:dateTime"/>
                <xsd:element minOccurs="0" name="reference" nillable="true"
        type="xsd:string"/>
                <xsd:element minOccurs="0" name="text" nillable="true"
        type="xsd:string"/>
                <xsd:element minOccurs="0" name="signType" nillable="true"
        type="type:signType"/>
```

```
        <xsd:element minOccurs="0" name="application" nillable="true"
    type="xsd:string"/>
        <xsd:element minOccurs="0" name="documentList" nillable="true"
    type="type:documentList"/>
        <xsd:element minOccurs="0" name="signLineList" nillable="true"
    type="type:signLineList"/>
        <xsd:element minOccurs="0" name="remitterList" nillable="true"
    type="type:remitterList"/>
        <xsd:element minOccurs="0" name="parameterList" nillable="true"
    type="type:parameterList"/>
        <xsd:element minOccurs="0" name="noticeList" nillable="true"
    type="type:noticeList"/>
        <xsd:element minOccurs="0" name="actionList" nillable="true"
    type="type:actionList"/>
        <xsd:element minOccurs="0" name="requestStatus" nillable="true"
    type="type:requestStatus"/>
    </xsd:sequence>
</xsd:complexType>
```

- **identifier:** Identificador de la solicitud. Lo genera el portafirmas cuando se crea la solicitud mendiante createRequest
- **subject:** Asunto de la solicitud. Campo obligatorio
- **fentry:** Fecha sin hora en la que la petición entra en el portafirmas
- **fstart:** Fecha a partir de la cual la petición es válida. La escribe el remitente
- **fexpiration:** Fecha a partir de la cual la petición deja de ser válida. La escribe el remitente
- **reference:** Campo que contiene una referencia que podrá ser vista por el firmante de la aplicación. Esta referencia no se puede usar para identificar la solicitud a través de los webservices. Hay que usar el campo identifier.
- **text:** Contenido del cuerpo del mensaje. Admite formato HTML.
- **signType:** Indica si la firma se realiza en paralelo o cascada. Los tipos de firma se definen en el punto 6.6
- **application:** Identificador de la aplicación que envía la solicitud. Para poder crear, modificar o leer una solicitud, la aplicación debe colgar de una aplicación padre que coincide con el nombre del usuario autenticado. Por ejemplo, si la petición la manda el usuario ACCEDA a la aplicación ACCEDA_TRAMITE1, debe existir en el portafirmas una aplicación llamada ACCEDA de la que cuelgue una subaplicación llamada ACCEDA_TRAMITE1.
- **importanceLevel:** Nivel de importancia de la petición. Se define en el punto
- **documentList:** Lista de documentos. El tipo documentList contiene varios elementos de tipo document, que representan los datos de un documento. El tipo document se define

en el punto 6.4. Cuando se recupera una petición, los elementos de tipo document no incluyen el contenido del mismo.

- **signLineList:** Lista de líneas de firma de la solicitud. Se define en el punto 6.7
- **remitterList:** Lista de remitentes de la solicitud. Se define en el punto 6.8
- **parameterList:** Lista de parámetros de la solicitud. Este campo existe en previsión de usos futuros, pero actualmente no se utiliza.
- **noticeList:** Indica los cambios de estado en los que se ha de enviar una notificación al remitente de la petición. Elelemento noticeList se define en punto 6.9.
- **actionList:** OPCIONAL. SIGNIFICADO DESCONOCIDO
- **requestStatus:** Campo de conveniencia devuelto por el portafirmas indicando el estado de la solicitud. Se define en el punto 6.13. Este campo se ignorará cuando se cree o actualice una petición
- **commentList**: Lista de comentarios que los usuarios han ido añadiendo a la petición. Se define en el punto 6.14
- **timestampInfo**: Información sobre si se quiere que la petición tenga sello de tiempo. Ver punto 6.15

Ejemplo de request:

```
<request>
	<identifier>12345678</identifier>
	<subject>Prueba</subect>
	<fentry>2010-10-26T21:32:52</fentry>
	<fstart>2010-10-26T21:32:52</fstart>
	<fexpiration>2010-10-26T21:32:52</fexpiration>
	<reference>Referencia Opcional</reference>
	<text>contenido del mensaje</text>
	<signType>CASCADA</signType>
	<application>ACCEDA_TRAMITE</application>
	<importanceLevel>
		<codeLevel>2</codeLevel>
		<description>Alta</description>
	</importanceLevel>
	<documentList>
		<document>
			<identifier>12345678</identifier>
			<name>documento de prueba.pdf</name>
			<mime>application/pdf</mime>
			<documentType>
				<identifier>INFORME</identifier>
				<description>Informe genérico</description>
				<valid>true</valid>
			</documentType>
```

```
            <content>[[CONTENIDO EN BASE64]]</content>
            <type/>
            <uri/>
            <sign>true</sign>
        </document>
    </documentList>
    <signLineList>
        <signLine>
            <signerList>
                <signer>
                    <user>
                        <identifier>12345678Z</identifier>
                        <name>Nombre</name>
                        <surname1>Apellido1</surname1>
                        <surname2>Apellido2</surname2>
                    </user>
                    <state>FIRMADO</state>
                    <fstate>2010-10-26T21:32:52</fstate>
                </signer>
            </signerList>
            <signLineType>VISTOBUENO</signLineType>
        </signLine>
        <signLine>
            <signerList>
                <signer>
                    <user>
                        <identifier>12345679A</identifier>
                        <name>BNombre</name>
                        <surname1>BApellido1</surname1>
                        <surname2>BApellido2</surname2>
                    </user>
                    <state>FIRMADO</state>
                    <fstate>2010-10-26T21:32:52</fstate>
                </signer>
            </signerList>
            <signLineType>FIRMA</signLineType>
        </signLine>
    </signLineList>
    <remitterList>
        <user>
            <identifier>ACCEDA</identifier>
            <name>ACCEDA</name>
            <surname1>ACCEDA</surname1>
            <surname2>ACCEDA</surname2>
        </user>
    </remitterList>
```

```
        <parameterList />
        <noticeList>
                <state>
                        <identifier>FIRMADO</identifier>
                </state>
        </noticeList>
        <actionList />
        <commentList>
                <comment>
                        <subject>COMENTARIO</subject>
                        <textComment>Todo correcto</textComment>
                        <fmodify>2010-10-26T21:31:52</fmodify>
                        <user>
                                <identifier>12345678Z</identifier>
                                <name>Nombre</name>
                                <surname1>Apellido1</surname1>
                                <surname2>Apellido2</surname2>
                        </user>
                </comment>
        </commentList>
        <requestStatus>ACEPTADA</requestStatus>
</request>
```

6.6 Tipo de firma (signType)

El tipo signType define el tipo de firma que se debe realizar sobre una petición. El tipo signType, responde al siguiente esquema XML:

```
<xsd:simpleType name="signType">
        <xsd:restriction base="xsd:string">
                <xsd:enumeration value="CASCADA"/>
                <xsd:enumeration value="PARALELA"/>
                <xsd:enumeration value="PRIMER FIRMANTE"/>
        </xsd:restriction>
</xsd:simpleType>
```

A continuación se definen los 3 tipos de firma:

- **Cascada:** En la firma en cascada, las diversas lineas de firma, se deben firmar o dar el visto bueno en el orden en que están en la petición. Si hubiera varias personas en una misma linea, lo podrá firmar/validar cualquiera de ellas.
- **Paralelo:** En la firma en paralelo, las diversas lineas de firma, se pueden firmar o dar el visto bueno en cualquier orden idependietemente de como estén en la petición. Si hubiera varias personas en una misma linea, lo podrá firmar/validar cualquiera de ellas. Si una persona estuviese

en más de una linea, deberá firmar/validar en primer lugar la primera de ellas, aunque eso no impide que otros usuarios puedan firmar otras lineas.

- **Primer firmante:** Este tipo de firma es igual que la firma en paralelo salvo que un usuario que esté en 2 lineas de firma, sólo podrá firmar una de ellas

Ejemplo:

```
<signType>CASCADA</signType>
```

6.7 Lista de líneas de firma (signLineList)

La lista de líneas de firma, nos indica el número de firmas o visto bueno que se debe dar a los documentos de la petición. Cada línea de firma puede contener varios firmantes, pero será firmada o aceptada exclusivamente por una persona.

```
<xsd:complexType name="signLineList">
	<xsd:sequence>
		<xsd:element maxOccurs="unbounded" minOccurs="0" name="signLine"
	type="type:signLine"/>
	</xsd:sequence>
</xsd:complexType>
```

La lista de líneas de firma está compuesta por una secuencia de varias líneas de firma representadas por el tipo signLine:

```
<xsd:complexType name="signLine">
	<xsd:sequence>
		<xsd:element minOccurs="0" name="signerList" nillable="true"
	type="type:signerList"/>
		<xsd:element minOccurs="0" name="type" nillable="true"
	type="type:signLineType"/>
	</xsd:sequence>
</xsd:complexType>
```

- **signerList:** Lista de personas y cargos que componen la linea de firma. Está descrito en el punto 6.2
- **signLineType:** Tipo de linea de firma. Acepta 2 valores:
  - **FIRMA:** La linea debe ser aceptada y firmada.
  - **VISTOBUENO:** La linea debe ser aceptada.

6.8 Lista de remitentes (remitterList)

El objeto remitterList nos permite indicar quienes son las personas que remiten una petición. Los remitentes deben ser usuarios válidos de la aplicación. Aunque el modelo de datos permite poner varios remitentes, la aplicación actualmente limita el número a 1 remitente, que además debe ser la misma aplicación que se pone como origen de la petición.

```
<xsd:complexType name="remitterList">
	<xsd:sequence>
		<xsd:element name="user" type="type:user" minOccurs="0"
maxOccurs="unbounded" />
	</xsd:sequence>
</xsd:complexType>
```

El elemento remitterList, se compone de una lista de elementos de tipo user que se define en el apartado 6.3. Los remitentes deben ser usuarios concreto. No pueden ser cargos.

6.9 Lista de avisos (noticeList)

El elemento noticeList incluye una lista de estados de la solicitud para los que el remitente de una solicitud desea recibir un aviso en el correo electrónico configurado en su aplicación.

```
<xsd:complexType name="noticeList">
	<xsd:sequence>
		<xsd:element maxOccurs="unbounded" minOccurs="0" name="state"
	type="type:state"/>
	</xsd:sequence>
</xsd:complexType>
```

6.10 Lista de estados (stateList, state)

El elemento stateList consiste en una lista de diferentes estados en los que puede hallarse un firmante dentro de una linea de firma

```
<xsd:complexType name="stateList">
	<xsd:sequence>
		<xsd:element maxOccurs="unbounded" minOccurs="0" name="state"
	type="type:state"/>
	</xsd:sequence>
</xsd:complexType>
```

Cada uno de esos estados viene representado por una cadena de texto

```
<xsd:complexType name="state">
	<xsd:sequence>
```

```
        <xsd:element minOccurs="0" name="identifier" nillable="true"
    type="xsd:string"/>
    </xsd:sequence>
</xsd:complexType>
```

Los valores posibles son:

- FIRMADO
- VISTOBUENO
- DEVUELTO
- CADUCADO
- EN ESPERA
- LEIDO
- NUEVO
- RETIRADO

6.11 Firma (signature)

El elemento signature representa un objeto con la información de un firma electrónica de un documento.

```
<xsd:complexType name="signature">
    <xsd:sequence>
        <xsd:element minOccurs="0" name="identifier" nillable="true"
    type="xsd:string"/>
        <xsd:element minOccurs="0" name="sign" nillable="true"
    type="xsd:boolean"/>
        <xsd:element minOccurs="0" name="signFormat" nillable="true"
    type="type:signFormat"/>
        <xsd:element minOccurs="0" name="content" nillable="true"
    type="xsd:base64Binary" xmime:expectedContentTypes="application/octet-
    stream"/>
    </xsd:sequence>
</xsd:complexType>
```

- **identifier:** Identificador del documento
- **sign:** Indica si el documento está o no firmado
- **signFormat:** Formato de la firma (PKCS7, CMS, CADES, XADES, XADES IMPLICITO, XADES EXPLICITO, XADES ENVELOPING, PDF)
- **content:** Contenido en base 64 de la firma

6.12 Nivel de importancia (importanceLevel)

El elemento importanceLevel representa un objeto con la información del nivel de importancia de una petición.

```
<xsd:complexType name="importanceLevel">
	<xsd:sequence>
		<xsd:element minOccurs="0" name="levelCode" nillable="false"
		type="xsd:string"/>
		<xsd:element minOccurs="0" name="description" nillable="false"
		type="xsd:string"/>
	</xsd:sequence>
</xsd:complexType>
```

- **levelCode:** Código numérico del nivel de importancia
- **description:** Descripción del nivel de importancia

6.13 Estado de la petición (requestStatus)

Este elemento sirve para representar el estado en que se encuentra una petición.

```
<xsd:simpleType name="requestStatus">
	<xsd:restriction base="xsd:string">
		<xsd:enumeration value="ACEPTADO" />
		<xsd:enumeration value="RECHAZADO" />
		<xsd:enumeration value="CADUCADO" />
		<xsd:enumeration value="EN PROCESO" />
		<xsd:enumeration value="RETIRADO" />
	</xsd:restriction>
</xsd:simpleType>
```

6.14 Lista de comentarios (commentList)

El elemento commentList consiste en una lista de los comentarios que los usuarios han ido añadiendo a la petición:

```
<xsd:complexType name="commentList">
	<xsd:sequence>
		<xsd:element name="comment" type="type:comment" minOccurs="0"
maxOccurs="unbounded"/>
	</xsd:sequence>
</xsd:complexType>
```

Cada uno de estos comentarios viene representado por el siguiente tipo:

```
<xsd:complexType name="comment">
	<xsd:sequence>
		<xsd:element name="subject" type="xsd:string" nillable="false"
minOccurs="0"/>
		<xsd:element name="textComment" type="xsd:string" nillable="false"
minOccurs="0"/>
		<xsd:element name="fmodify" type="xsd:dateTime" nillable="false"
minOccurs="0"/>
		<xsd:element name="user" type="type:userJob" nillable="false"
minOccurs="0"/>
	</xsd:sequence>
</xsd:complexType>
```

- **subject:** Asunto del comentario.
- **textComment:** Texto del comentario.
- **fmodify**: Fecha de última modificación.
- **user**: Usuario que escribió el comentario.

6.15 Información sello de tiempo (timestampInfo)

Contiene información sobre si se quiere que la firma de los documentos lleve sello de tiempo. No es obligatorio, si no se envía se entenderá que no.

```
<xsd:complexType name="timestampInfo">
	<xsd:sequence>
		<xsd:element name="addTimestamp" type="xsd:boolean" nillable="false"
minOccurs="0"/>
	</xsd:sequence>
</xsd:complexType>
```

- **addTimestamp:** Indica si se quiere que se le ponga sello de tiempo a la firma o no.

6.16 Parametros (parameter)

Contiene un par *nombre parámetro – valor*.

```
<xsd:complexType name="parameter">
	<xsd:sequence>
		<xsd:element name="identifier" type="xsd:string" nillable="true"
minOccurs="0"/>
```

```
            <xsd:element name="value" type="xsd:string" nillable="true"
minOccurs="0"/>
        </xsd:sequence>
</xsd:complexType>
```

- **identifier:** Identificador del parámetro. En esta versión el único valor admitido es i*dentificador.ldap* cuando la lista de parámetros forme parte del tipo *enhancedUserJobInfo* de un *enhancedUser*.
- **value:** Valor del parámetro.

6.17 Lista de parámetros (parameterList)

Lista de parámetros:

```
<xsd:complexType name="parameterList">
        <xsd:sequence>
            <xsd:element name="parameter" type="type:parameter" minOccurs="0"
maxOccurs="unbounded" />
        </xsd:sequence>
</xsd:complexType>
```

- **parameter:** Ver punto 6.16.

6.18 Sede (seat)

Contiene información de una sede de Portafirmas.

```
<xsd:complexType name="seat">
        <xsd:sequence>
            <xsd:element name="code" type="xsd:string" nillable="true"
minOccurs="0"/>
            <xsd:element name="description" type="xsd:string" nillable="true"
minOccurs="0"/>
        </xsd:sequence>
</xsd:complexType>
```

- **code:** Código identificador de la sede.
- **description:** Descripción de la sede**.**

6.19 Lista de sedes (seatList)

Tipo que representa una lista de sedes de Portafirmas.

```
<xsd:complexType name="seatList">
	<xsd:sequence>
		<xsd:element name="seat" type="type:seat" minOccurs="0"
maxOccurs="unbounded" />

	</xsd:sequence>
</xsd:complexType>
```

- **seat**: Ver 6.18

6.20 Información ampliada de usuario y cargo (enhancedUserJobInfo)

Contiene información sobre la sede y la vigencia del usuario o el cargo.

```
<xsd:complexType name="enhancedUserJobInfo">
	<xsd:sequence>
		<xsd:element name="seat" type="type:seat" nillable="true"
minOccurs="0"/>
		<xsd:element name="valid" type="xsd:boolean" nillable="true"
minOccurs="0"/>
		<xsd:element name="visibleOtherSeats" type="xsd:boolean" nillable="true"
minOccurs="0"/>
		<xsd:element name="parameterList" type="type:parameterList"
nillable="true" minOccurs="0"/>
	</xsd:sequence>
</xsd:complexType>
```

- **seat:** Sede del usuario o el cargo. Ver 6.18
- **valid:** True si el usuario o cargo está vigente, false en caso contrario.
- **visibleOtherSeats**: True si el usuario es visible para usuarios de otras sedes.
- **parameterList**: Lista de parámetros asociados al usuario o al cargo. Ver punto 6.17

6.21 Lista de usuarios con información ampliada (enhancedUserList)

El elemento **enhancedUserList** es una lista de usuarios con información ampliada:

```
<xsd:complexType name="enhancedUserList">
	<xsd:sequence>
		<xsd:element name="enhancedUser" type="type:enhancedUser"
minOccurs="0" maxOccurs="unbounded" />
	</xsd:sequence>
```

```
</xsd:complexType>
```

El tipo **enhancedUser** es el siguiente:

```
<xsd:complexType name="enhancedUser">
	<xsd:sequence>
		<xsd:element name="user" type="type:user" nillable="true"
minOccurs="0"/>
		<xsd:element name="enhancedUserJobInfo"
type="type:enhancedUserJobInfo" nillable="true" minOccurs="0"/>
		<xsd:element name="identificadorLdap" type="xsd:string" nillable="true"
minOccurs="0"/>
	</xsd:sequence>
</xsd:complexType>
```

- **user:** Contiene información del usuario. Ver 6.3
- **enhancedUserJobInfo:** Contiene información de la sede y la vigencia del usuario y otros parámetros. Ver 6.20

6.22 Lista de cargos con información ampliada (enhancedJobList)

El elemento **enhancedJobList** es una lista de cargos con información ampliada:

```
<xsd:complexType name="enhancedJobList">
	<xsd:sequence>
		<xsd:element name="enhancedJob" type="type:enhancedJob" minOccurs="0"
maxOccurs="unbounded" />
	</xsd:sequence>
</xsd:complexType>
```

El tipo **enhancedJob** es el siguiente:

```
<xsd:complexType name="enhancedJobr">
	<xsd:sequence>
		<xsd:element name="job" type="type:job" nillable="true" minOccurs="0"/>
		<xsd:element name="enhancedUserJobInfo"
type="type:enhancedUserJobInfo" nillable="true" minOccurs="0"/>
	</xsd:sequence>
</xsd:complexType>
```

- **job:** Contiene información del cargo. Ver 6.3
- **enhancedUserJobInfo:** Contiene información de la sede y la vigencia del cargo. Ver 6.20

6.23 Asociación entre usuario y cargo (enhancedUserJobAssociated)

Representa la asociación entre un usuario y cargos asociados y viceversa.

```
<xsd:complexType name="enhancedUserJobAssociated">
	<xsd:sequence>
		<xsd:element name="enhancedUser" type="type:enhancedUser" />
		<xsd:element name="enhancedJob" type="type:enhancedJob" />
		<xsd:element name="fstart" type="xsd:dateTime" />
		<xsd:element name="fend" type="xsd:dateTime" nillable="true"
minOccurs="0"/>
	</xsd:sequence>
</xsd:complexType>
```

- **enhancedUser:** Contiene información ampliada del usuario. Ver 6.21
- **enhancedJob:** Contiene información ampliada del cargo. Ver 6.22
- **fstart**: Fecha de inicio de la relación usuario-cargo.
- **fend**: Fecha de fin de la relación usuario-cargo. Si es nula, se considera que la relación no tiene tiempo máximo de vigencia.

6.24 Lista de asociaciones entre usuario y cargo (enhancedUserJobAssociatedList)

Lista de asociaciones entre un usuario y cargos asociados y viceversa.

```
<xsd:complexType name="enhancedUserJobAssociatedList">
	<xsd:sequence>
		<xsd:element name="enhancedUserJobAssociated"
type="type:enhancedUserJobAssociated" minOccurs="0" maxOccurs="unbounded" />
	</xsd:sequence>
</xsd:complexType>
```

- **enhancedUserJobAssociated:** Ver 6.23

7 Servicio de respuesta para aplicaciones que se integran con Portafirmas.

Opcionalmente, las aplicaciones que se integran con Portafirmas pueden implementar un servicio de respuesta, de manera que cuando una petición enviada a Portafirmas por dichas aplicaciones sea firmada, rechazada o retirada, Portafirmas hará una llamada al servicio de respuesta de la aplicación que ha enviado la petición en cuestión, con los datos actualizados de la petición.

Cada una de las aplicaciones que desee integrar esta funcionalidad deberá implementar un WS que siga la especificación del WSDL AdviceService.wsdl que sigue a continuación:

```
<?xml version='1.0' encoding='UTF-8'?><wsdl:definitions name="AdviceService"
targetNamespace="urn:juntadeandalucia:cice:pfirma:advice:v2.0"
xmlns:pfirma="urn:juntadeandalucia:cice:pfirma:advice:v2.0"
xmlns:request="urn:juntadeandalucia:cice:pfirma:advice:request:v2.0"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:type="urn:juntadeandalucia:cice:pfirma:type:v2.0"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">

  <wsdl:types>

<xsd:schema attributeFormDefault="unqualified" elementFormDefault="unqualified"
targetNamespace="urn:juntadeandalucia:cice:pfirma:advice:request:v2.0"
xmlns:pfirma="urn:juntadeandalucia:cice:pfirma:advice:v2.0"
xmlns:request="urn:juntadeandalucia:cice:pfirma:advice:request:v2.0"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:type="urn:juntadeandalucia:cice:pfirma:type:v2.0"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">


                    <xsd:import namespace="urn:juntadeandalucia:cice:pfirma:type:v2.0"
schemaLocation="WebServiceTypes.xsd"/>


                    <xsd:element name="updateRequestStatus">

                        <xsd:complexType>

                            <xsd:sequence>

                                <xsd:element name="authentication"
type="type:authentication"/>
```

```
                                        <xsd:element name="request"
type="type:request"/>
                                        <xsd:element maxOccurs="unbounded"
minOccurs="0" name="signature" type="type:signature"/>
                                    </xsd:sequence>
                                </xsd:complexType>
                            </xsd:element>

                            <xsd:element name="updateRequestStatusResponse">
                                <xsd:complexType>
                                    <xsd:sequence>
                                        <xsd:element minOccurs="0" name="result"
type="xsd:boolean"/>
                                    </xsd:sequence>
                                </xsd:complexType>
                            </xsd:element>
                        </xsd:schema>
  </wsdl:types>
  <wsdl:message name="updateRequestStatus">
    <wsdl:part element="request:updateRequestStatus" name="updateRequestStatus">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="updateRequestStatusResponse">
    <wsdl:part element="request:updateRequestStatusResponse"
name="updateRequestStatusResponse">
```

```
  </wsdl:part>
 </wsdl:message>
 <wsdl:message name="pfirmaException">
  <wsdl:part element="type:exceptionInfo" name="pfirmaException">
  </wsdl:part>
 </wsdl:message>
 <wsdl:portType name="AdviceService">
  <wsdl:operation name="updateRequestStatus">
   <wsdl:input message="pfirma:updateRequestStatus" name="updateRequestStatus">
  </wsdl:input>
   <wsdl:output message="pfirma:updateRequestStatusResponse"
name="updateRequestStatusResponse">
  </wsdl:output>
   <wsdl:fault message="pfirma:pfirmaException" name="pfirmaException">
  </wsdl:fault>
  </wsdl:operation>
 </wsdl:portType>
 <wsdl:binding name="AdviceServiceSoapBinding" type="pfirma:AdviceService">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="updateRequestStatus">
   <soap:operation soapAction="" style="document"/>
   <wsdl:input name="updateRequestStatus">
    <soap:body use="literal"/>
   </wsdl:input>
```

```
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="pfirmaException">
      <soap:fault name="pfirmaException" use="literal"/>
    </wsdl:fault>
  </wsdl:operation>
 </wsdl:binding>
 <wsdl:service name="AdviceService">
  <wsdl:port binding="pfirma:AdviceServiceSoapBinding" name="AdviceServicePort">
    <soap:address location="http://servidor/AdviceService"/>
  </wsdl:port>
 </wsdl:service>
</wsdl:definitions>
```

La configuración del servicio de respuesta de cada aplicación se hace en el menú de *Administración*, en la pestaña *Aplicaciones*.

Habrá que añadir los siguientes parámetros:

- **RESPUESTA.WS.USUARIO:** Usuario que usará la aplicación Portafirmas para autenticarse en la aplicación destino.
- **RESPUESTA.WS.PASSWORD**: Password que usará la aplicación Portafirmas para autenticarse en la aplicación destino.
- **RESPUESTA.WS.ACTIVA**: Valores posibles S/N, indica si la respuesta está activa, es decir, si hay que invocar al servicio de respuesta de la aplicación.
- **RESPUESTA.WS.WSDLLOCATION**: URL del WSDL del servicio de respuesta de la aplicación.
- **RESPUESTA.WS.NOTIFINTERMEDIOS**: Indica si hay que avisar a la aplicación destino no sólo cuando las peticiones estén finalizadas, si no cada vez que alguno de los destinatarios de una petición realice alguna acción de firma o visto bueno.