Consejería de Economía y Hacienda
Dirección General de Patrimonio, Informática y Telecomunicaciones

MANUAL DEL PROGRAMADOR DE LA APLICACIÓN DE INFORMES.

Soporte al framework de desarrollo Javato (fase II) Murcia

Octubre de 2013

ÍNDICE

Tabla de ilustraciones

1. INTRODUCCIÓN

El presente documento es el manual del programador para la aplicación de informes.

En primer lugar se verán el servicio web que proporciona la aplicación de informes. Este servicio proporciona una utilidad para obtener informes partiendo de ciertos orígenes de datos y plantillas.

El origen de datos del informe puede ser un fichero xml o un origen de datos que ya existe en la aplicación de informes.

La plantilla puede ser de birt, odt o una plantilla que ya esté dada de alta en la aplicación de informes.

Para terminar con los servicios web, habrá un último apartado en el que se muestra el fichero *wsdl* con la descripción del servicio web.

Finalmente, se mostrará cómo lanzar un informe invocándolo por parámetro.

2. INVOCACIÓN POR WEB SERVICES

2.1 Elementos y Estructuras de Datos involucradas

2.1.1. TEMPLATEDATA

Esta clase representa las plantillas que se pasan al servicio web. Tiene dos atributos:

- **String templateType**: es un *String* que indica el tipo de plantilla. Para establecer y leer el valor se utilizarán los métodos setTemplateType y getTemplateType respectivamente. Los valores que puede tomar son:
  - ***birt***
  - ***odt***
- **DataHandler templateFile*:*** Encapsula el fichero que representa la plantilla. Para establecer y leer el valor se utilizarán los métodos setTemplateFile y getTemplateFile respectivamente.

2.1.2. REPORTPARAM

Esta clase representa los parámetros que se le pueden pasar a los informes. Tiene 2 atributos:

- **String key**: cadena con el nombre del parámetro. Para establecer y leer su valor se utilizará setKey y getKey, respectivamente.
- **Object value**: objeto con el valor del parámetro. Para establecer y leer su valor se utilizarán los métodos setValue y getValue, respectivamente.

2.1.3. CONTENT TYPES

Los *content types* que se pueden emplear para la obtención de informes son:

- Para los archivos ODT: **application/vnd.oasis.opendocument.text**
- Para los archivos PDF: **application/pdf**
- Para los archivos DOC: **application/msword**
- Para los archivos RTF: **text/rtf**
- Para los archivos HTML: **text/html**

2.2 Interfaz del Servicio Web

La interfaz del servicio web 'ReportWS' se puede ver en la Figura 1.

```
@WebService(name = "ReportWS", serviceName="ReportService", targetNamespace =
"http://webservice.informes.javato.carm.es/1.0/")
public interface ReportWS {

  @WebMethod
  public @XmlMimeType("application/octet-stream")DataHandler getReport(
    @WebParam(name = "templateData") TemplateData templateData,
    @WebParam(name = "xmlDataSource")@XmlMimeType("application/octet-stream")
    DataHandler xmlDataSource,
    @WebParam(name = "reportApp") String reportApp,
    @WebParam(name = "contentType") String contentType
  );

  @WebMethod
  public @XmlMimeType("application/octet-stream") DataHandler getReportByIdentifier(
    @WebParam(name = "reportIdentifier") String reportIdentifier,
```

```
    @WebParam(name = "reportApp") String reportApp,
    @WebParam(name = "reportParamList") List<ReportParam> reportParamList,
    @WebParam(name = "contentType") String contentType
  );

  @WebMethod
  public @XmlMimeType("application/octet-stream") DataHandler getReportByTemplate(
    @WebParam(name = "templateIdentifier") String templateIdentifier,
    @WebParam(name = "templateApp") String templateApp,
    @WebParam(name = "xmlDataSource")@XmlMimeType("application/octet-stream")
    DataHandler xmlDataSource,
    @WebParam(name = "contentType") String contentType
  );

  @WebMethod
  public @XmlMimeType("application/octet-stream") DataHandler getReportByDataSource(
    @WebParam(name = "templateData") TemplateData templateData,
    @WebParam(name = "dataSourceIdentifier") String dataSourceIdentifier,
    @WebParam(name = "dataSourceApp") String dataSourceApp,
    @WebParam(name = "reportParamList") List<ReportParam> reportParamList,
    @WebParam(name = "contentType") String contentType
  );

  @WebMethod
  public @XmlMimeType("application/octet-stream")DataHandler
    getReportByIdentifierWithOrder(
    @WebParam(name = "reportIdentifier") String reportIdentifier,
    @WebParam(name = "reportApp") String reportApp,
    @WebParam(name = "reportParamList") List<Object> reportParamList,
    @WebParam(name = "contentType") String contentType
  );

  @WebMethod
  public @XmlMimeType("application/octet-stream") DataHandler
    getReportByDataSourceWithOrder(
    @WebParam(name = "templateData") TemplateData templateData,
    @WebParam(name = "dataSourceIdentifier") String dataSourceIdentifier,
    @WebParam(name = "dataSourceApp") String dataSourceApp,
    @WebParam(name = "reportParamList") List<Object> reportParamList,
    @WebParam(name = "contentType") String contentType
  );

}
```

Figura 1. Interfaz del servicio web 'ReportWS'

Este servicio tiene 4 métodos para la obtención de un informe. Estos métodos se describen a continuación:

- **getReport**: método al que se le pasan 4 parámetros necesarios para la obtención de informe y devuelve un *DataHandler* que lo contiene. En este caso, tanto la plantilla como el origen da datos se proporcionan a la aplicación, sin que esté ninguna de ellas dada de alta previamente.
    - **templateData**, que es del tipo *TemplateData* explicado en el apartado 2.1.1. Representa la plantilla que indica cómo se mostrarán los datos del informe.
    - **xmlDataSource**, que es DataHandler que representa un fichero xml con los datos del origen.
    - **reportApp:** Nombre de la aplicación a la que pertenece el informe.
    - **contentType**, que es una cadena de caracteres que indica el *content type* del informe devuelto. Los valores que se pueden emplear son los expuestos en el apartado 2.1.3.
- **getReportByIdentifier**: método al que se le pasan 4 parámetros para la obtención del informe y devuelve un *DataHandler* que lo contiene. En este caso, el informe está ya creado en la aplicación y se referencia por su identificador.
    - **reportIdentifier**, que es de tipo String y representa el identificador del informe (que ya debe estar dado de alta en la aplicación).
    - **reportApp:** Nombre de la aplicación a la que pertenece el informe.

- **reportParamList:** representa la lista de parámetros que puede necesitar el informe. La estructura de cada parámetro está en el apartado 2.1.2.
- **contentType**, que es una cadena de caracteres que indica el *content type* del informe devuelto. Los valores que se pueden emplear son los expuestos en el apartado 2.1.3.

- **getReportByTemplate**: método al que se le pasan 4 parámetros para la obtención del informe y devuelve un *DataHandler* que lo contiene. En este caso, la plantilla está ya creada en la aplicación y se referencia por su identificador.
  - **templateIdentifier**: cadena de caracteres que indica el identificador de la plantilla.
  - **templateApp:** Nombre de la aplicación a la que pertenece la plantilla.
  - **xmlDataSource**, que es un *DataHandler* que representa un fichero xml con los datos del origen.
  - **contentType**, que es una cadena de caracteres que indica el *content type* del informe devuelto. Los valores que se pueden emplear son los expuestos en el apartado 2.1.3.
- **getReportByDataSource**: método al que se le pasan 5 parámetros para la obtención del informe y devuelve un *DataHandler* que lo contiene. En este caso, el origen de datos está ya creado en la aplicación y se referencia por su identificador.
  - **templateData**, que es del tipo *TemplateData* explicado en el apartado 2.1.1. Representa la plantilla que indica cómo se mostrarán los datos del informe.
  - **dataSourceIdentifier**, identificador con el que se ha registrado en la aplicación el origen de datos.
  - **dataSourceApp:** Nombre de la aplicación a la que pertenece el origen de datos.
  - **reportParamList:** representa la lista de parámetros que puede necesitar el informe. La estructura de cada parámetro está en el apartado 2.1.2.
  - **contentType**, que es una cadena de caracteres que indica el *content type* del informe devuelto. Los valores que se pueden emplear son los expuestos en el apartado 2.1.3.
- **getReportByIdentifierWithOrder**: método al que se le pasan 4 parámetros para la obtención del informe y devuelve un *DataHandler* que lo contiene. En este caso, el informe está ya creado en la aplicación y se referencia por su nombre.
  - **reportIdentifier**, que es de tipo String y representa el identificador del informe (que ya debe estar dado de alta en la aplicación).
  - **reportApp:** Nombre de la aplicación a la que pertenece el informe.
  - **reportParamList:** Contiene una lista de valores que representan los valores de los parámetros del informe, ordenados según el orden especificado en el diseño del informe.
  - **contentType**, que es una cadena de caracteres que indica el *content type* del informe devuelto. Los valores que se pueden emplear son los expuestos en el apartado 2.1.3.
- **getReportByDataSourceWithOrder**: método al que se le pasan 5 parámetros para la obtención del informe y devuelve un *DataHandler* que lo contiene. En este caso, el origen de datos está ya creado en la aplicación y se referencia por su identificador.
  - **templateData**, que es del tipo *TemplateData* explicado en el apartado 2.1.1. Representa la plantilla que indica cómo se mostrarán los datos del informe.
  - **dataSourceIdentifier**, identificador con el que se ha registrado en la aplicación el origen de datos.
  - **dataSourceApp:** Nombre de la aplicación a la que pertenece el origen de datos.
  - **reportParamList:** Contiene una lista de valores que representan los valores de los parámetros del origen, ordenados según el orden especificado en el diseño del origen de datos.

    contentType, que es una cadena de caracteres que indica el *content type* del informe

2.3 Descripción del Servicio Web (wsdl)

El fichero *wsdl* para este servicio se puede obtener en la URL "http/s://[servidor]:[puerto]/jInformes/webservice/1.0/ReportService?wsdl" del servidor correspondiente en función del entorno (desarrollo, pruebas, producción). El contenido de la *wsdl* se puede ver en la Figura 2

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="ReportService"
 targetNamespace="http://webservice.informes.javato.carm.es/1.0/"
 xmlns:ns1="http://schemas.xmlsoap.org/soap/http"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:tns="http://webservice.informes.javato.carm.es/1.0/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <wsdl:types>
  <xs:schema elementFormDefault="unqualified"
   targetNamespace="http://webservice.informes.javato.carm.es/1.0/"
   version="1.0" xmlns:tns="http://webservice.informes.javato.carm.es/1.0/"
   xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:element name="getReport" type="tns:getReport" />
   <xs:element name="getReportByDataSource" type="tns:getReportByDataSource" />
   <xs:element name="getReportByDataSourceResponse"
type="tns:getReportByDataSourceResponse" />
   <xs:element name="getReportByDataSourceWithOrder"
type="tns:getReportByDataSourceWithOrder" />
   <xs:element name="getReportByDataSourceWithOrderResponse"
    type="tns:getReportByDataSourceWithOrderResponse" />
   <xs:element name="getReportByIdentifier" type="tns:getReportByIdentifier" />
   <xs:element name="getReportByIdentifierResponse"
type="tns:getReportByIdentifierResponse" />
   <xs:element name="getReportByIdentifierWithOrder"
type="tns:getReportByIdentifierWithOrder" />
   <xs:element name="getReportByIdentifierWithOrderResponse"
    type="tns:getReportByIdentifierWithOrderResponse" />
   <xs:element name="getReportByTemplate" type="tns:getReportByTemplate" />
   <xs:element name="getReportByTemplateResponse"
type="tns:getReportByTemplateResponse" />
   <xs:element name="getReportResponse" type="tns:getReportResponse" />
   <xs:complexType name="getReportByIdentifierWithOrder">
    <xs:sequence>
     <xs:element minOccurs="0" name="reportIdentifier" type="xs:string" />
     <xs:element minOccurs="0" name="reportApp" type="xs:string" />
     <xs:element maxOccurs="unbounded" minOccurs="0"
      name="reportParamList" type="xs:anyType" />
     <xs:element minOccurs="0" name="contentType" type="xs:string" />
    </xs:sequence>
   </xs:complexType>
   <xs:complexType name="getReportByIdentifierWithOrderResponse">
    <xs:sequence>
     <xs:element minOccurs="0" name="return"
      ns1:expectedContentTypes="application/octet-stream" type="xs:base64Binary"
      xmlns:ns1="http://www.w3.org/2005/05/xmlmime" />
    </xs:sequence>
   </xs:complexType>
   <xs:complexType name="getReportByDataSource">
    <xs:sequence>
     <xs:element minOccurs="0" name="templateData" type="tns:templateData" />
     <xs:element minOccurs="0" name="dataSourceIdentifier"
      type="xs:string" />
     <xs:element minOccurs="0" name="dataSourceApp" type="xs:string" />
     <xs:element maxOccurs="unbounded" minOccurs="0"
      name="reportParamList" type="tns:reportParam" />
     <xs:element minOccurs="0" name="contentType" type="xs:string" />
    </xs:sequence>
   </xs:complexType>
   <xs:complexType name="templateData">
    <xs:sequence>
     <xs:element minOccurs="0" name="templateFile"
```

```
      ns2:expectedContentTypes="application/octet-stream" type="xs:base64Binary"
      xmlns:ns2="http://www.w3.org/2005/05/xmlmime" />
     <xs:element minOccurs="0" name="templateType" type="xs:string" />
    </xs:sequence>
   </xs:complexType>
   <xs:complexType name="reportParam">
    <xs:sequence>
     <xs:element minOccurs="0" name="key" type="xs:string" />
     <xs:element minOccurs="0" name="value" type="xs:anyType" />
    </xs:sequence>
   </xs:complexType>
   <xs:complexType name="getReportByDataSourceResponse">
    <xs:sequence>
     <xs:element minOccurs="0" name="return"
      ns3:expectedContentTypes="application/octet-stream" type="xs:base64Binary"
      xmlns:ns3="http://www.w3.org/2005/05/xmlmime" />
    </xs:sequence>
   </xs:complexType>
   <xs:complexType name="getReport">
    <xs:sequence>
     <xs:element minOccurs="0" name="templateData" type="tns:templateData" />
     <xs:element minOccurs="0" name="xmlDataSource"
      ns4:expectedContentTypes="application/octet-stream" type="xs:base64Binary"
      xmlns:ns4="http://www.w3.org/2005/05/xmlmime" />
     <xs:element minOccurs="0" name="reportApp" type="xs:string" />
     <xs:element minOccurs="0" name="contentType" type="xs:string" />
    </xs:sequence>
   </xs:complexType>
   <xs:complexType name="getReportResponse">
    <xs:sequence>
     <xs:element minOccurs="0" name="return"
      ns5:expectedContentTypes="application/octet-stream" type="xs:base64Binary"
      xmlns:ns5="http://www.w3.org/2005/05/xmlmime" />
    </xs:sequence>
   </xs:complexType>
   <xs:complexType name="getReportByDataSourceWithOrder">
    <xs:sequence>
     <xs:element minOccurs="0" name="templateData" type="tns:templateData" />
     <xs:element minOccurs="0" name="dataSourceIdentifier"
      type="xs:string" />
     <xs:element minOccurs="0" name="dataSourceApp" type="xs:string" />
     <xs:element maxOccurs="unbounded" minOccurs="0"
      name="reportParamList" type="xs:anyType" />
     <xs:element minOccurs="0" name="contentType" type="xs:string" />
    </xs:sequence>
   </xs:complexType>
   <xs:complexType name="getReportByDataSourceWithOrderResponse">
    <xs:sequence>
     <xs:element minOccurs="0" name="return"
      ns6:expectedContentTypes="application/octet-stream" type="xs:base64Binary"
      xmlns:ns6="http://www.w3.org/2005/05/xmlmime" />
    </xs:sequence>
   </xs:complexType>
   <xs:complexType name="getReportByTemplate">
    <xs:sequence>
     <xs:element minOccurs="0" name="templateIdentifier"
      type="xs:string" />
     <xs:element minOccurs="0" name="templateApp" type="xs:string" />
     <xs:element minOccurs="0" name="xmlDataSource"
      ns7:expectedContentTypes="application/octet-stream" type="xs:base64Binary"
      xmlns:ns7="http://www.w3.org/2005/05/xmlmime" />
     <xs:element minOccurs="0" name="contentType" type="xs:string" />
    </xs:sequence>
   </xs:complexType>
   <xs:complexType name="getReportByTemplateResponse">
    <xs:sequence>
     <xs:element minOccurs="0" name="return"
      ns8:expectedContentTypes="application/octet-stream" type="xs:base64Binary"
      xmlns:ns8="http://www.w3.org/2005/05/xmlmime" />
    </xs:sequence>
   </xs:complexType>
   <xs:complexType name="getReportByIdentifier">
    <xs:sequence>
     <xs:element minOccurs="0" name="reportIdentifier" type="xs:string" />
```

```
      <xs:element minOccurs="0" name="reportApp" type="xs:string" />
      <xs:element maxOccurs="unbounded" minOccurs="0"
       name="reportParamList" type="tns:reportParam" />
      <xs:element minOccurs="0" name="contentType" type="xs:string" />
     </xs:sequence>
    </xs:complexType>
    <xs:complexType name="getReportByIdentifierResponse">
     <xs:sequence>
      <xs:element minOccurs="0" name="return"
       ns9:expectedContentTypes="application/octet-stream" type="xs:base64Binary"
       xmlns:ns9="http://www.w3.org/2005/05/xmlmime" />
     </xs:sequence>
    </xs:complexType>
   </xs:schema>
  </wsdl:types>
  <wsdl:message name="getReportByIdentifierWithOrderResponse">
   <wsdl:part element="tns:getReportByIdentifierWithOrderResponse"
    name="parameters">
   </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getReportResponse">
   <wsdl:part element="tns:getReportResponse" name="parameters">
   </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getReportByDataSource">
   <wsdl:part element="tns:getReportByDataSource" name="parameters">
   </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getReportByIdentifierResponse">
   <wsdl:part element="tns:getReportByIdentifierResponse" name="parameters">
   </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getReportByIdentifierWithOrder">
   <wsdl:part element="tns:getReportByIdentifierWithOrder"
    name="parameters">
   </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getReportByIdentifier">
   <wsdl:part element="tns:getReportByIdentifier" name="parameters">
   </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getReportByDataSourceWithOrderResponse">
   <wsdl:part element="tns:getReportByDataSourceWithOrderResponse"
    name="parameters">
   </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getReport">
   <wsdl:part element="tns:getReport" name="parameters">
   </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getReportByTemplateResponse">
   <wsdl:part element="tns:getReportByTemplateResponse" name="parameters">
   </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getReportByTemplate">
   <wsdl:part element="tns:getReportByTemplate" name="parameters">
   </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getReportByDataSourceWithOrder">
   <wsdl:part element="tns:getReportByDataSourceWithOrder"
    name="parameters">
   </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getReportByDataSourceResponse">
   <wsdl:part element="tns:getReportByDataSourceResponse" name="parameters">
   </wsdl:part>
  </wsdl:message>
  <wsdl:portType name="ReportWS">
   <wsdl:operation name="getReportByIdentifierWithOrder">
    <wsdl:input message="tns:getReportByIdentifierWithOrder"
     name="getReportByIdentifierWithOrder">
    </wsdl:input>
    <wsdl:output message="tns:getReportByIdentifierWithOrderResponse"
     name="getReportByIdentifierWithOrderResponse">
```

```
    </wsdl:output>
   </wsdl:operation>
   <wsdl:operation name="getReportByDataSource">
    <wsdl:input message="tns:getReportByDataSource" name="getReportByDataSource">
    </wsdl:input>
    <wsdl:output message="tns:getReportByDataSourceResponse"
     name="getReportByDataSourceResponse">
    </wsdl:output>
   </wsdl:operation>
   <wsdl:operation name="getReport">
    <wsdl:input message="tns:getReport" name="getReport">
    </wsdl:input>
    <wsdl:output message="tns:getReportResponse" name="getReportResponse">
    </wsdl:output>
   </wsdl:operation>
   <wsdl:operation name="getReportByDataSourceWithOrder">
    <wsdl:input message="tns:getReportByDataSourceWithOrder"
     name="getReportByDataSourceWithOrder">
    </wsdl:input>
    <wsdl:output message="tns:getReportByDataSourceWithOrderResponse"
     name="getReportByDataSourceWithOrderResponse">
    </wsdl:output>
   </wsdl:operation>
   <wsdl:operation name="getReportByTemplate">
    <wsdl:input message="tns:getReportByTemplate" name="getReportByTemplate">
    </wsdl:input>
    <wsdl:output message="tns:getReportByTemplateResponse"
     name="getReportByTemplateResponse">
    </wsdl:output>
   </wsdl:operation>
   <wsdl:operation name="getReportByIdentifier">
    <wsdl:input message="tns:getReportByIdentifier" name="getReportByIdentifier">
    </wsdl:input>
    <wsdl:output message="tns:getReportByIdentifierResponse"
     name="getReportByIdentifierResponse">
    </wsdl:output>
   </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="ReportServiceSoapBinding" type="tns:ReportWS">
   <soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http" />
   <wsdl:operation name="getReportByIdentifierWithOrder">
    <soap:operation soapAction="" style="document" />
    <wsdl:input name="getReportByIdentifierWithOrder">
     <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="getReportByIdentifierWithOrderResponse">
     <soap:body use="literal" />
    </wsdl:output>
   </wsdl:operation>
   <wsdl:operation name="getReportByDataSource">
    <soap:operation soapAction="" style="document" />
    <wsdl:input name="getReportByDataSource">
     <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="getReportByDataSourceResponse">
     <soap:body use="literal" />
    </wsdl:output>
   </wsdl:operation>
   <wsdl:operation name="getReport">
    <soap:operation soapAction="" style="document" />
    <wsdl:input name="getReport">
     <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="getReportResponse">
     <soap:body use="literal" />
    </wsdl:output>
   </wsdl:operation>
   <wsdl:operation name="getReportByDataSourceWithOrder">
    <soap:operation soapAction="" style="document" />
    <wsdl:input name="getReportByDataSourceWithOrder">
     <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output name="getReportByDataSourceWithOrderResponse">
```

```
      <soap:body use="literal" />
     </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="getReportByTemplate">
     <soap:operation soapAction="" style="document" />
     <wsdl:input name="getReportByTemplate">
      <soap:body use="literal" />
     </wsdl:input>
     <wsdl:output name="getReportByTemplateResponse">
      <soap:body use="literal" />
     </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="getReportByIdentifier">
     <soap:operation soapAction="" style="document" />
     <wsdl:input name="getReportByIdentifier">
      <soap:body use="literal" />
     </wsdl:input>
     <wsdl:output name="getReportByIdentifierResponse">
      <soap:body use="literal" />
     </wsdl:output>
    </wsdl:operation>
   </wsdl:binding>
   <wsdl:service name="ReportService">
    <wsdl:port binding="tns:ReportServiceSoapBinding" name="ReportWSPort">
     <soap:address
      location="http://javatodkxp/jInformes/webservice/1.0/ReportService" />
    </wsdl:port>
   </wsdl:service>
</wsdl:definitions>
```

Figura 2. Fichero wsdl

3. INVOCACIÓN DE INFORMES POR PARÁMETRO

También es posible lanzar un informe mediante invocación de un *servlet*. El *servlet* se puede llamar mediante la una URL como la siguiente:

http/s://<dirección_servidor>[:<puerto>]/jInformes/pages/reportLauncher?informeIdentifier=<identificador del informe>&informeApp=<nombre de la aplicación>&backUrl=<url de vuelta>&perfil=<nombre del perfil>&grupo=<código del grupo>&ambito=<código del ámbito>

Los parámetros que hay que especificar son los siguientes:

- **informeIdentifier** que se corresponde con el identificador del informe
- **informeApp** que es el nombre de la aplicación del informe
- **backUrl'** que no es obligatorio, pero si se especifica será a la dirección que reenvíe al pulsar el botón 'Volver'
- **perfil**: Nombre del perfil. Si se indica este parámetro se usará como valor para los parámetros de sistema de tipo perfil.
- **grupo**: Código del grupo. Si se indica algún valor se usará como valor para los parámetros de sistema de tipo grupo.
- **ambito**: Código del ámbito. Si se indica algún valor, se usará como valor para los parámetros de sistema de tipo ambito.

En el caso de que se invoque el lanzamiento de un informe mediante el *servlet*, si el usuario no ha hecho previamente *login* en alguna aplicación del SSO de la CARM, la aplicación redirigirá al usuario a una pantalla de autenticación.

Además, si la aplicación requiere trabajar con selectores (perfil, grupo, ámbito) y estos no han sido informados a través de los parámetros correspondientes se mostrará la pantalla de selección de selectores.

Si el usuario no tiene permisos para lanzar el informe se le mostrará una pantalla como la de la Figura 3.

Figura 3. Pantalla que indica que el usuario no tiene permisos para lanzar el informe

En caso de que el usuario sí tenga permisos para lanzar el informe, y suponiendo que queremos lanzar el informe, la pantalla que visualizará será similar a la de lanzamiento de la aplicación pero sin el menú y sin dar la opción de cambiar de aplicación (en la cabecera, el lugar en que se muestra la aplicación no será un enlace para poder modificar), tal y como se puede ver en la Figura 4.

En el supuesto de que no se hubiera especificado nada en el parámetro 'backUrl' no aparecería el botón Volver.

Figura 4. Pantalla de lanzamiento desde el servlet.

4. LISTADO DE INFORMES

Es posible invocar la pantalla del listado de informes por medio de la siguiente URL (sustituyendo los datos entre corchetes por el valor adecuado):

http/s://[nombre_servidor]:[puerto]/jInformes/pages/reportSelect?informeApp=[nombreAplicación]&backUrl=[url_retorno]&backMenu=[texto_mostrar]&perfil=[perfil_filtro]&ambito=[ambito_filtro]&grupo=[grupo_filtro]

Donde los parámetros que se le indican son:

- **informeApp**: nombre de la aplicación con la que se desea acceder a la aplicación de informes. Es el único parámetro obligatorio.
- **backUrl**: URL a la que se desea volver.
- **backMenu**: texto que se mostrará junto a 'Salir'. Al pulsar sobre él reenviará a la url que se indique en *backUrl* (ver Figura 5).
- **perfil**: Nombre del perfil por el que se desea filtrar. Si se indica algún valor, filtrará el listado con los informes que tengan ese perfil en sus opciones de seguridad.
- **grupo**: Código del grupo por el que se desea filtrar. Si se indica algún valor, filtrará el listado con los informes que tengan ese grupo en sus opciones de seguridad.
- **ámbito**: Código del ámbito por el que se desea filtrar. Si se indica algún valor, filtrará el listado con los informes que tengan ese ámbito (además del perfil que se indique) en sus opciones de seguridad. No se permitirá indicar un ámbito sin perfil.

En el caso de que el usuario no se hubiera autenticado en alguna de las aplicaciones del SSO de la CARM, la aplicación reenviará a la página de *login* del servidor CAS.

Además, si la aplicación requiere trabajar con selectores (perfil, grupo, ámbito) y estos no han sido informados a través de los parámetros correspondientes se mostrará la pantalla de selección de selectores.

En cualquier caso, una vez que el usuario se ha autenticado, se mostrará una pantalla como la de la Figura 5.

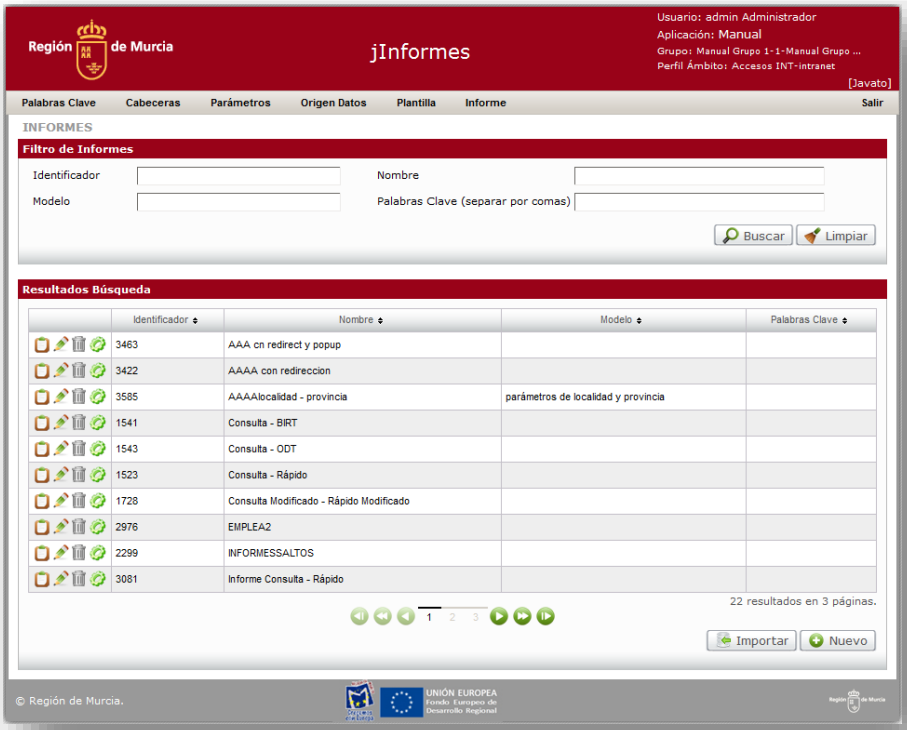

Figura 5. Listado de informes