DISEÑO TÉCNICO PLUGIN LOGIN

30 de enero de 2016

Contrato: P.O.166.14

Ports de Balears

Autoritat Portuària de Balears

Control de versiones del documento

Control de Cambios			
Fecha	**Autor**	**Versión**	**Cambios**
2016/01/30	INDRA	v1.0	Versión inicial

Revisado por		
Nombre	**Fecha**	**Área, departamento o empresa**

Aprobado por		
Nombre	**Fecha**	**Área, departamento o empresa**

Lista de distribución		
Nombre	**Área, departamento o empresa**	**Correo electrónico**

Índice

1. Objeto del documento

En este documento se detalla el diseño técnico de la integración de SISTRA para soportar las funcionalidades requeridas para el login en la implantación de la APB:

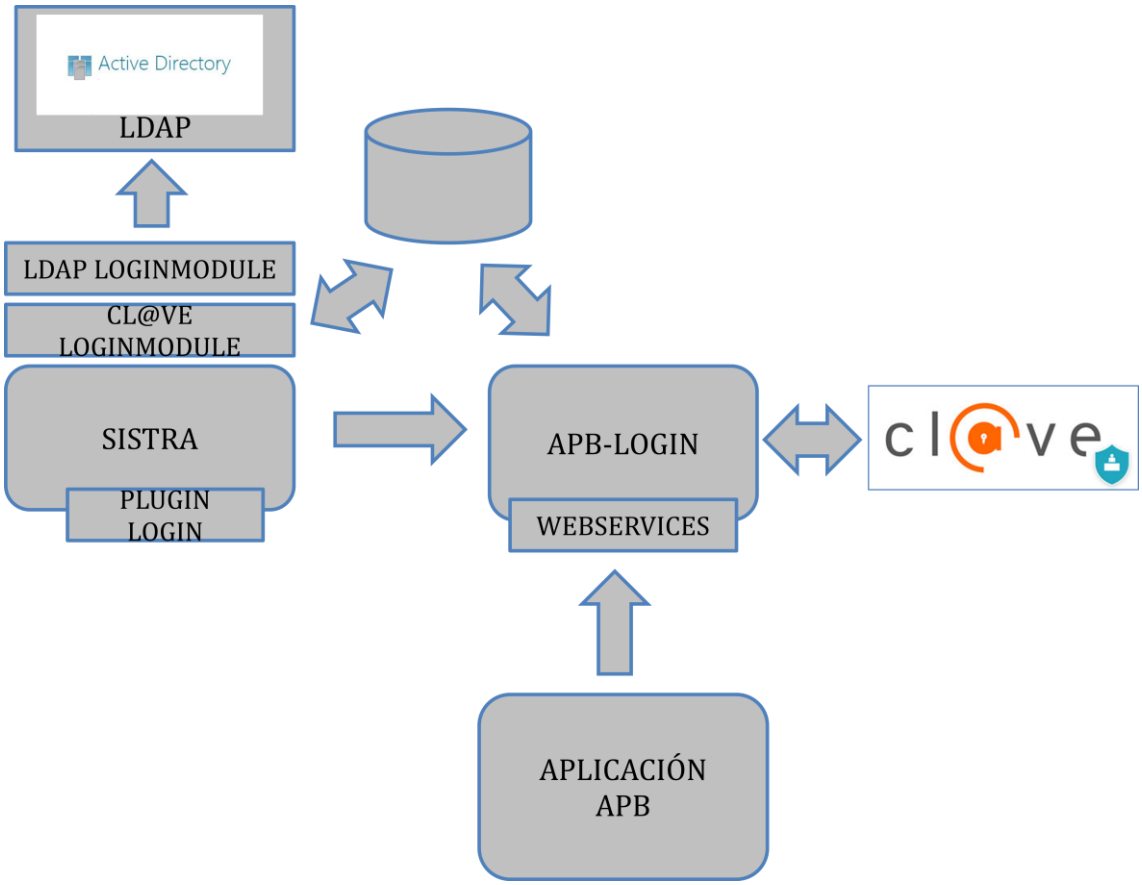
- Integración con el LDAP corporativo para la autenticación en los Backoffices
- Integración con Cl@ve para la autenticación en los Frontales (asistente tramitación y zona personal)

2. Introducción

Para la integración con Cl@ve se ha desarrollado un módulo externo para realizar la integración. Este módulo se ha ampliado para permitir que otras aplicaciones distintas a SISTRA puedan utilizar Cl@ve (p.e. aplicación de Consejos). De esta forma cualquier aplicación en la APB que requiera el uso de Cl@ve puede utilizar este módulo para realizar la integración.

3. Arquitectura

En el siguiente diagrama se muestran los elementos que intervienen en la solución:



Donde se distinguen los siguientes elementos:

- LDAP: LDAP corporativo de la APB, que es Active Directory.
- Cl@ve: Plataforma común del Sector Público Administrativo Estatal para la identificación, autenticación y firma electrónica mediante el uso de claves concertadas, abierta a su utilización por parte de todas las Administraciones Públicas.
- Aplicación APB: Aplicación APB que quiere integrarse con Cl@ve (p.e. Consejeros)
- SISTRA: Instalación de SISTRA para la APB.
- Plugin de login: plugin que implementa el interfaz definido por SISTRA que permite obtener los datos del usuario a partir del Principal.
- LoginModules: Módulos del estándar JAAS que se encargan de realizar la autenticación. Se configuran a nivel de JBoss. Se han desarrollado 2 LoginModule:
    - LDAP LoginModule: se encarga de validar un usuario/password contra el LDAP corporativo.
    - Cl@ve LoginModule: se encarga de validar un ticket generado por el módulo APB Login tras haber realizado el login con Cl@ve.
- APB Login: Módulo que se encarga de realizar el proceso de login contra Cl@ve y genera un ticket para obtener la información del login. Tiene 2 modos de funcionamiento:

- Login para SISTRA: el intercambio de información se realiza a través de la BBDD.
- Login para otras aplicaciones APB: el intercambio de información se realiza a través de webservices.

4. Modelo de datos

El modelo de datos utilizado por la solución es el siguiente:

ZPE_TICKET			
TCK_CODIGO	NUMBER(20)	<pk>	not null
TCK_FCSES	DATE		not null
TCK_IDIOMA	VARCHAR2(2)		not null
TCK_IDPS	VARCHAR2(100)		null
TCK_URLCBK	VARCHAR2(4000)		not null
TCK_URLDST	VARCHAR2(4000)		not null
TCK_TICKET	VARCHAR2(100)		null
TCK_FCALTA	DATE		null
TCK_NIVAUT	VARCHAR2(1)		null
TCK_NIF	VARCHAR2(10)		null
TCK_NOMAPE	VARCHAR2(1000)		null
TCK_FCULT	DATE		null

ZPE_TICKEX			
TCX_CODIGO	NUMBER(20)	<pk>	not null
TCX_FCSES	DATE		not null
TCX_IDIOMA	VARCHAR2(2)		not null
TCX_IDPS	VARCHAR2(100)		not null
TCX_URLCBK	VARCHAR2(4000)		not null
TCX_TICKET	VARCHAR2(100)		null
TCX_FCALTA	DATE		null
TCX_NIVAUT	VARCHAR2(50)		null
TCX_NIF	VARCHAR2(10)		null
TCX_NOM	VARCHAR2(1000)		null
TCX_APE	VARCHAR2(1000)		null

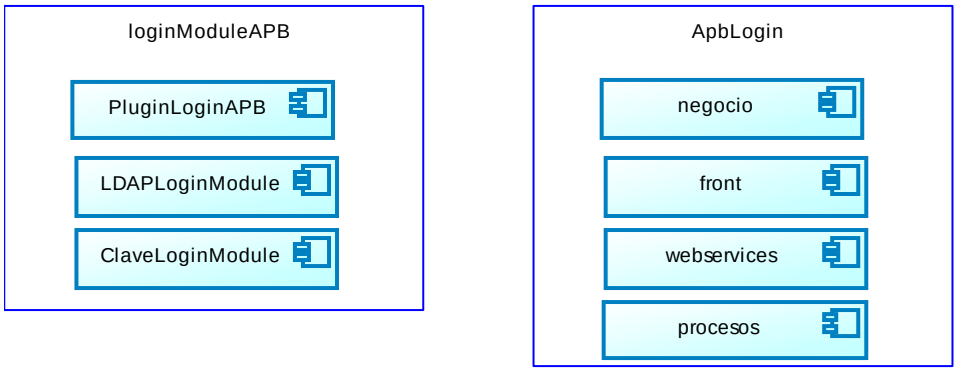
ZPE_TICKET	
Sesiones para autenticaciones basadas en ticket para Cl@ve para SISTRA	
CAMPO	DESCRIPCIÓN
TCK_CODIGO	Código interno secuencial
TCK_FCSES	Fecha inicio sesión
TCK_IDIOMA	Idioma
TCK_IDPS	IDPS
TCK_URLCBK	Url callback sistra
TCK_URLDST	Url destino sistra
TCK_TICKET	Ticket
TCK_FCALTA	Fecha alta
TCK_NIVAUT	Nivel autenticación (C/U)
TCK_NIF	Nif
TCK_NOMAPE	Nombre completo
TCK_FCULT	Fecha ultimo login

ZPE_TICKEX	
Sesiones para autenticaciones basadas en ticket para Cl@ve para aplicaciones externas	
CAMPO	DESCRIPCIÓN
TCX_CODIGO	Código interno secuencial
TCX_FCSES	Fecha inicio sesion
TCX_IDIOMA	Idioma
TCX_IDPS	Idps (separados por ;)
TCX_URLCBK	Url callback
TCX_TICKET	Ticket
TCX_FCALTA	Fecha alta
TCX_NIVAUT	Nivel autenticación (Idp con el que se ha autenticado)
TCX_NIF	Nif
TCX_NOM	Nombre
TCX_APE	Apellidos

Las tablas se han ubicado en el esquema del módulo ZONAPER.

5. Módulos del sistema

Los módulos a desarrollar serán los siguientes:



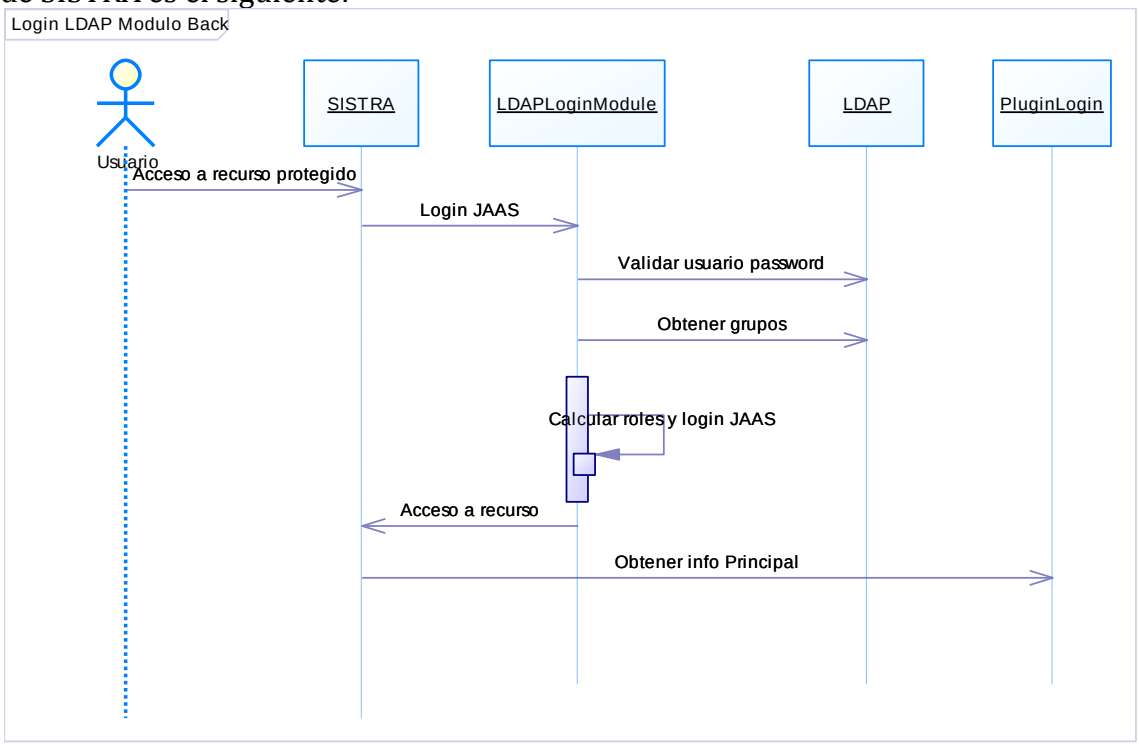
- loginModuleAPB: Jar que contendrá:
  - PluginLoginAPB: plugin login APB
  - LDAPLoginModule: login module para validar usuarios contra LDAP
  - ClaveLoginModule: login module para validar usuarios contra tickets generados por APB Login en autenticaciones contra Cl@ve.

- ApbLogin: contendrá wars de frontal y de webservices
  - negocio: lógica de negocio de integración con Cl@ve
  - front: capa web de apb login que se encarga de las redirecciones entre aplicaciones
  - webservices: capa de servicios web para integración con aplicaciones APB
  - procesos: procesos de background (purgado tickets, etc.)

6. Diagrama de funcionamiento

1. Proceso de login para SISTRA con LDAP

El diagrama de secuencia para la autenticación mediante LDAP para los módulos de back de SISTRA es el siguiente:

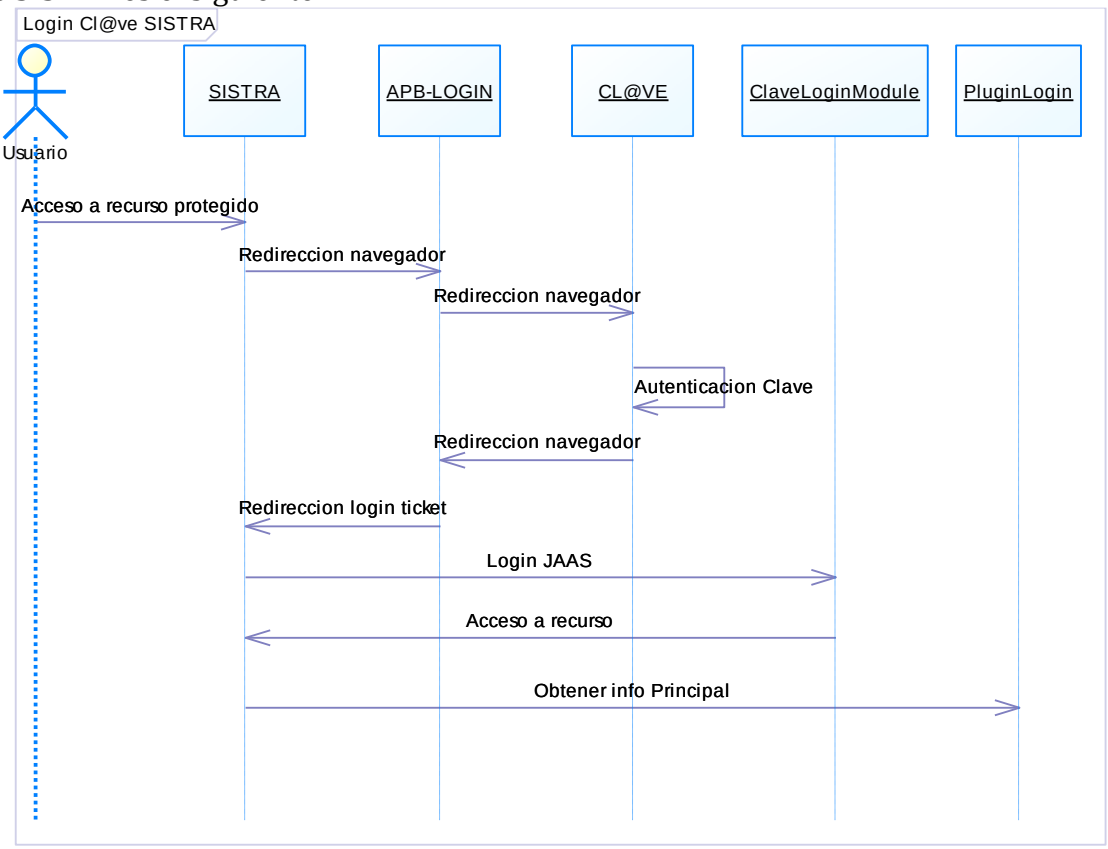


Los pasos son:

1) El usuario accede a un recurso protegido del módulo de back
2) Se procede con la autenticación JAAS y como estos módulos están configurados para autenticación Basic se solicita al usuario el usuario y password. El usuario y password es validado por el módulo LDAPLoginModule, para ello consulta al LDAP y se obtienen los datos del usuario y se obtienen los grupos a los que pertenece. Con esta información se crea el principal correspondiente (del tipo ApbPrincipal).
3) Una vez validado el usuario por el LoginModule se permite el acceso al recurso
4) Para obtener la información del usuario autenticado se accede al plugin de login que permite obtener la información del principal correspondiente (ApbLogin).

2. Proceso de login para SISTRA con Cl@ve

El diagrama de secuencia para la autenticación mediante Cl@ve para los módulos de front de SISTRA es el siguiente:



Los pasos son:

1. El usuario accede a un recurso protegido del módulo de front y se redirige a la página de login, que muestre la opción de autenticarse con Cl@ve.
2. Al pulsar la opción de autenticarse con Cl@ve se redirige el navegador al módulo APB Login pasando en la petición por POST los siguientes parámetros:
   - urlCallbackLogin: url a la que se debe redirigir tras finalizar el proceso de autenticación con Cl@ve. A esta url se le pasará por POST un parámetro con un ticket para poder obtener la información de autenticación.
   - urlDestino: url del recurso protegido al que se ha intentado acceder y al que se debe redirigir tras la autenticación.
   - metodos: métodos de autenticación soportados (Certificado / Usuario). El mapeo a los tipos usados por Cl@ve son:
     i. Certificado: aFirma
     ii. Usuario: AEAT y SS
   - Idioma: idioma

El módulo APB Login genera sesión de autenticación y almacena en la tabla de BBDD la información pasada en estos parámetros.

3. El módulo APB Login prepara la petición de autenticación a Cl@ve y la firma. En esta petición se indica la url de retorno que debe utilizar Cl@ve para retornar al módulo APB Login. Se redirige el navegador del usuario a Cl@ve pasando por POST la petición de autenticación.
4. Cl@ve procede a autenticar al usuario según el método seleccionado.
5. Cl@ve redirige el navegador a la url de retorno pasando por POST un parámetro con la respuesta firmada, que contiene los datos de la autenticación. El módulo APB Login procesa la respuesta de Cl@ve y con los datos de la autenticación se actualiza en la tabla de BBDD los datos de la autenticación generando un ticket.
6. El módulo APB Login redirige a la url de callback de login pasando el ticket. Esta url de callback procede a autenticar via JAAS utilizando como password el ticket pasado.
7. Se procede con la autenticación JAAS con el ClaveLoginModule y se valida el ticket contra la tabla de BBDD. Con esta información se crea el principal correspondiente (del tipo ApbPrincipal).
8. Una vez validado el usuario por el LoginModule se permite el acceso al recurso
9. Para obtener la información del usuario autenticado se accede al plugin de login que permite obtener la información del principal correspondiente (ApbLogin).

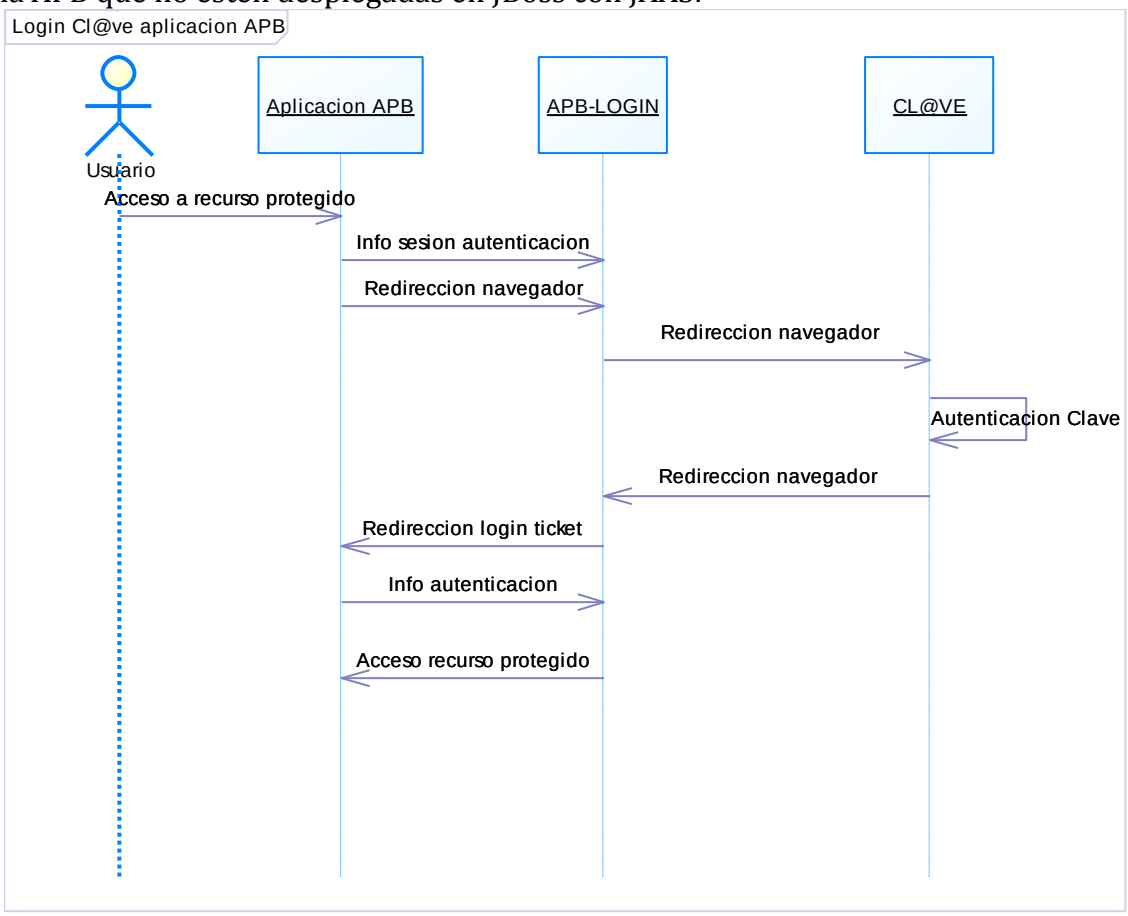
Hay que indicar que para el proceso de login en Sistra se mantiene la información del ticket en la tabla ya que tal como funciona el login JAAS periódicamente tras cumplirse cierto timeout se necesita volver a comprobar la autenticación. Se han establecido controles de seguridad para:

- establecer timeout desde que se inicia el proceso de autenticación
- establecer timeout desde que se realiza la autenticación hasta que se valida el ticket
- establecer timeout desde que se ha validado por última vez el ticket
- evitar que un ticket pueda ser reusado por otra sesión

La tabla de tickets se purga periódicamente para borrar tickets expirados.

3. Proceso de login para aplicación APB con Cl@ve

El diagrama de secuencia para la autenticación mediante Cl@ve para otras aplicaciones de la APB que no estén desplegadas en JBoss con JAAS:



Los pasos son:

1) El usuario accede a un recurso protegido de la aplicación y se redirige a la página de login, que muestre la opción de autenticarse con Cl@ve.
2) Al pulsar la opción de autenticarse con Cl@ve, se invoca al servicio web de APB Login para pasar la información de la sesión de autenticación:
   - urlCallbackLogin: url a la que se debe redirigir tras finalizar el proceso de autenticación con Cl@ve. A esta url se le pasará por POST un parámetro con un ticket para poder obtener la información de autenticación.
   - metodos: métodos de autenticación soportados (aFirma, AEAT y SS)
   - Idioma: idioma

   El módulo APB Login genera sesión de autenticación y almacena en la tabla de BBDD la información pasada en estos parámetros. Como respuesta al webservice se indica la url a la que se debe redirigir el navegador.

3) La aplicación redirige el navegador a la url obtenida como resultado de la invocación al webservice.
4) El módulo APB Login prepara la petición de autenticación a Cl@ve y la firma. En esta petición se indica la url de retorno que debe utilizar Cl@ve para retornar al módulo APB Login. Se redirige el navegador del usuario a Cl@ve pasando por POST la petición de autenticación.
5) Cl@ve procede a autenticar al usuario según el método seleccionado.
6) Cl@ve redirige el navegador a la url de retorno pasando por POST un parámetro con la respuesta firmada, que contiene los datos de la autenticación. El módulo APB Login procesa la respuesta de Cl@ve y con los datos de la autenticación se actualiza en la tabla de BBDD los datos de la autenticación generando un ticket.
7) El módulo APB Login redirige a la url de callback de login pasando el ticket.
8) La aplicación accede al webservice de APB Login para obtener la información de autenticación a partir del ticket. Una vez usado el ticket, este se borra de la tabla para evitar que pueda ser usado otra vez. Se procede con la autenticación y se valida el usuario en la aplicación.
9) Una vez validado el usuario permite el acceso al recurso

7. ANEXO 1: Servicio web APB-LOGIN

```
<?xml version='1.0' encoding='UTF-8'?>
<wsdl:definitions xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="urn:es:apb:login:ws:v1:login"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:ns1="http://schemas.xmlsoap.org/soap/http" name="LoginService_v1_00"
targetNamespace="urn:es:apb:login:ws:v1:login">
  <wsdl:types>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:tns="urn:es:apb:login:ws:v1:login" attributeFormDefault="unqualified"
elementFormDefault="unqualified" targetNamespace="urn:es:apb:login:ws:v1:login">
<xs:element name="iniciarSesionRequest" type="tns:iniciarSesionRequest"/>
<xs:element name="iniciarSesionResponse" type="tns:iniciarSesionResponse"/>
<xs:element name="ticketRequest" type="tns:ticketRequest"/>
<xs:element name="ticketResponse" type="tns:ticketResponse"/>
<xs:complexType name="iniciarSesionRequest">
<xs:sequence>
<xs:element minOccurs="0" name="peticion" type="tns:peticionIniciarSesion"/>
</xs:sequence>
</xs:complexType>
<xs:complexType final="extension restriction" name="peticionIniciarSesion">
<xs:sequence>
<xs:element name="urlCallbackLogin" type="xs:string"/>
<xs:element name="metodos" type="xs:string"/>
<xs:element name="idioma" type="xs:string"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="iniciarSesionResponse">
<xs:sequence>
<xs:element minOccurs="0" name="respuesta" type="tns:respuestaIniciarSesion"/>
</xs:sequence>
</xs:complexType>
<xs:complexType final="extension restriction" name="respuestaIniciarSesion">
<xs:sequence>
<xs:element name="urlRedireccion" type="xs:string"/>
</xs:sequence>
</xs:complexType>
<xs:complexType final="extension restriction" name="wPropiedadesError">
<xs:sequence>
<xs:element maxOccurs="unbounded" minOccurs="0" name="propiedadError"
nillable="true" type="tns:wPropiedadError"/>
</xs:sequence>
</xs:complexType>
<xs:complexType final="extension restriction" name="wPropiedadError">
<xs:sequence>
<xs:element minOccurs="0" name="propiedad" type="xs:string"/>
<xs:element minOccurs="0" name="valor" type="xs:string"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="ticketRequest">
<xs:sequence>
<xs:element minOccurs="0" name="peticion" type="tns:peticionTicket"/>
</xs:sequence>
</xs:complexType>
<xs:complexType final="extension restriction" name="peticionTicket">
<xs:sequence>
<xs:element name="ticket" type="xs:string"/>
```

```
</xs:sequence>
</xs:complexType>
<xs:complexType name="ticketResponse">
<xs:sequence>
<xs:element minOccurs="0" name="respuesta" type="tns:respuestaTicket"/>
</xs:sequence>
</xs:complexType>
<xs:complexType final="extension restriction" name="respuestaTicket">
<xs:sequence>
<xs:element name="nivelAutenticacion" type="xs:string"/>
<xs:element name="nif" type="xs:string"/>
<xs:element name="nombre" type="xs:string"/>
<xs:element minOccurs="0" name="apellidos" type="xs:string"/>
</xs:sequence>
</xs:complexType>
<xs:element name="ExcepcionWS" type="tns:ExcepcionWS"/>
<xs:complexType name="ExcepcionWS">
<xs:sequence>
<xs:element name="codigoError" nillable="true" type="xs:string"/>
<xs:element name="mensajeError" nillable="true" type="xs:string"/>
<xs:element name="detalleError" nillable="true" type="xs:string"/>
<xs:element                name="propiedadesError"                nillable="true"
type="tns:wPropiedadesError"/>
</xs:sequence>
</xs:complexType>
</xs:schema>
  </wsdl:types>
  <wsdl:message name="obtenerDatosTicketResponse">
    <wsdl:part element="tns:ticketResponse" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="iniciarSesion">
    <wsdl:part element="tns:iniciarSesionRequest" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="WException">
    <wsdl:part element="tns:ExcepcionWS" name="WException">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="iniciarSesionResponse">
    <wsdl:part element="tns:iniciarSesionResponse" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="obtenerDatosTicket">
    <wsdl:part element="tns:ticketRequest" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:portType name="LoginWebService">
    <wsdl:operation name="iniciarSesion">
      <wsdl:input message="tns:iniciarSesion" name="iniciarSesion">
    </wsdl:input>
      <wsdl:output                              message="tns:iniciarSesionResponse"
name="iniciarSesionResponse">
    </wsdl:output>
      <wsdl:fault message="tns:WException" name="WException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="obtenerDatosTicket">
      <wsdl:input message="tns:obtenerDatosTicket" name="obtenerDatosTicket">
```

```
    </wsdl:input>
      <wsdl:output                          message="tns:obtenerDatosTicketResponse"
name="obtenerDatosTicketResponse">
    </wsdl:output>
      <wsdl:fault message="tns:WException" name="WException">
    </wsdl:fault>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="LoginService_v1_00SoapBinding" type="tns:LoginWebService">
    <soap:binding                                                  style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="iniciarSesion">
      <soap:operation soapAction="" style="document"/>
      <wsdl:input name="iniciarSesion">
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output name="iniciarSesionResponse">
        <soap:body use="literal"/>
      </wsdl:output>
      <wsdl:fault name="WException">
        <soap:fault name="WException" use="literal"/>
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="obtenerDatosTicket">
      <soap:operation soapAction="" style="document"/>
      <wsdl:input name="obtenerDatosTicket">
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output name="obtenerDatosTicketResponse">
        <soap:body use="literal"/>
      </wsdl:output>
      <wsdl:fault name="WException">
        <soap:fault name="WException" use="literal"/>
      </wsdl:fault>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="LoginService_v1_00">
    <wsdl:port                          binding="tns:LoginService_v1_00SoapBinding"
name="LoginWebServiceImplPort">
      <soap:address                     location="http://localhost:28080/apb-login-
ws/LoginService"/>
    </wsdl:port>
  </wsdl:service>
   </wsdl:definitions>
```