VICEPRESIDÈNCIA I CONSELLERIA INNOVACIÓ, RECERCA I TURISME
DIRECCIÓ GENERAL DESENVOLUPAMENT TECNOLÒGIC

una manera de hacer europa

Fondo Europeo de Desarrollo Regional

Unió Europea

LOGINIB: Manual d'integració

14 de novembre de 2019

Serveis d'Administració Electrònica en el Govern de les Illes Balears

Lot 2 (Serveis electrònics per a la ciutadania)

Oficina Tècnica de Direcció de Projecte

Control de versions del document

Control de Canvis			
Data	**Autor**	**Versió**	**Canvis**
12/12/2018	Indra	v1.0	Versió inicial
14/11/2019	Indra	v1.0	Revisió documentació

Revisat per		
Nom	**Data**	**Àrea, departament o empresa**

Aprovat per		
Nom	**Data**	**Àrea, departament o empresa**

Llista de distribució		
Nom	**Àrea, departament o empresa**	**Correu electrònic**

Índex

1. Objecte

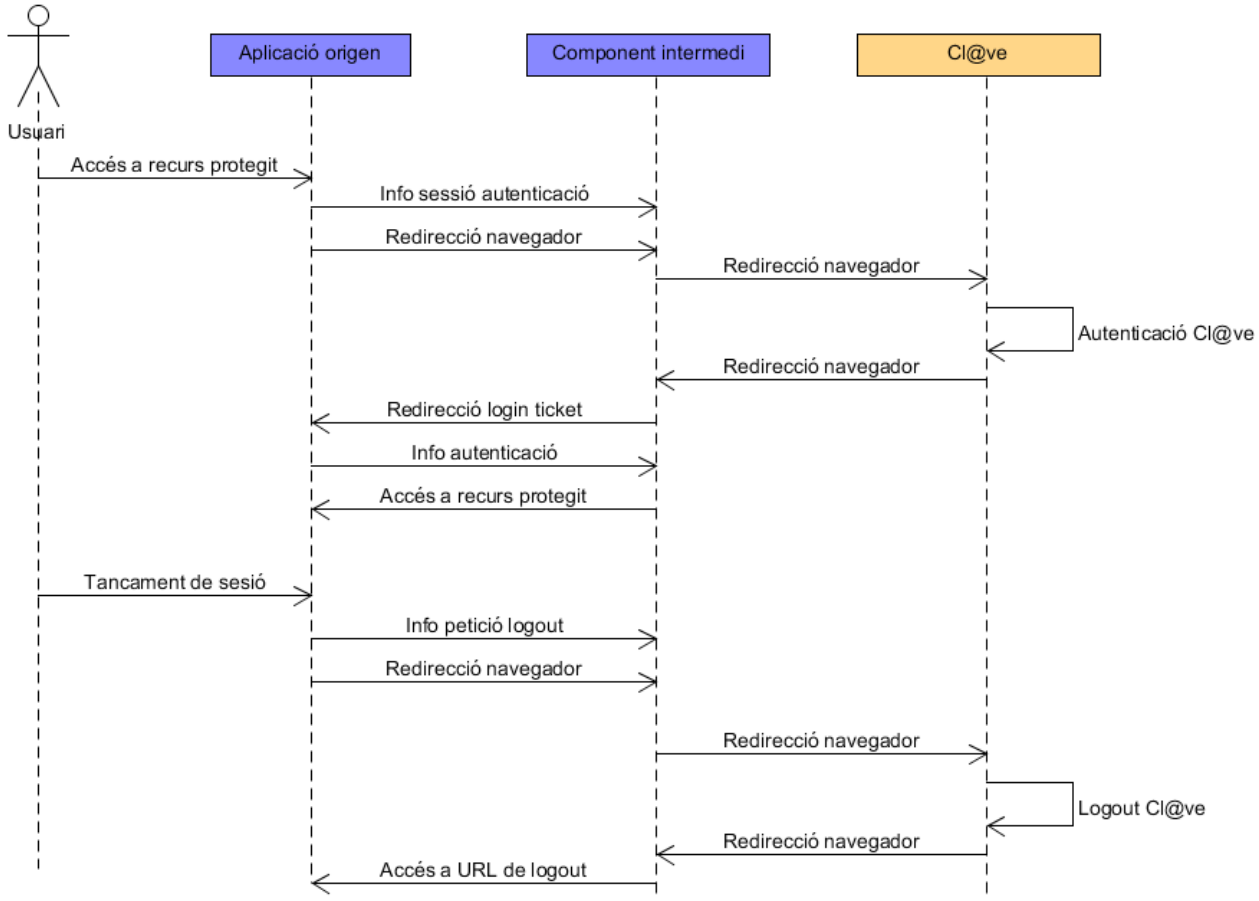
En aquest document s'aporta tota la informació necessària perquè qualsevol aplicació es pugui integrar amb el component horitzontal d'autenticació **LoginIB**.

2. Diagrama de funcionament

El component horitzontal d'autenticació LoginIB proveu, a qualsevol aplicació, les funcionalitats d'identificació i autenticació en base a tiquets d'un sol ús. Aquest component inclou la gestió de la pàgina de *login* i permet la utilització de diferents tipus d'autenticació. El tipus d'autenticació suportats actualment són Cl@ve (http://clave.gob.es/clave_Home/clave.html) i accés anònim, però en un futur es podrien incorporar nous sistemes d'autenticació.

El component horitzontal d'autenticació recollirà les peticions d'identificació per part de l'aplicació client i li traslladarà la informació d'autenticació mitjançant un tiquet d'accés.

A nivell de funcionament, el diagrama de seqüencia en el que se basa la funcionalitat del component és el que se presenta a continuació:



1. L'usuari accedeix a un recurs protegit de l'aplicació consumidora dels serveis, i en aquest moment l'aplicació ha d'invocar al servei web del component LoginIB per passar la informació de la sessió d'autenticació:

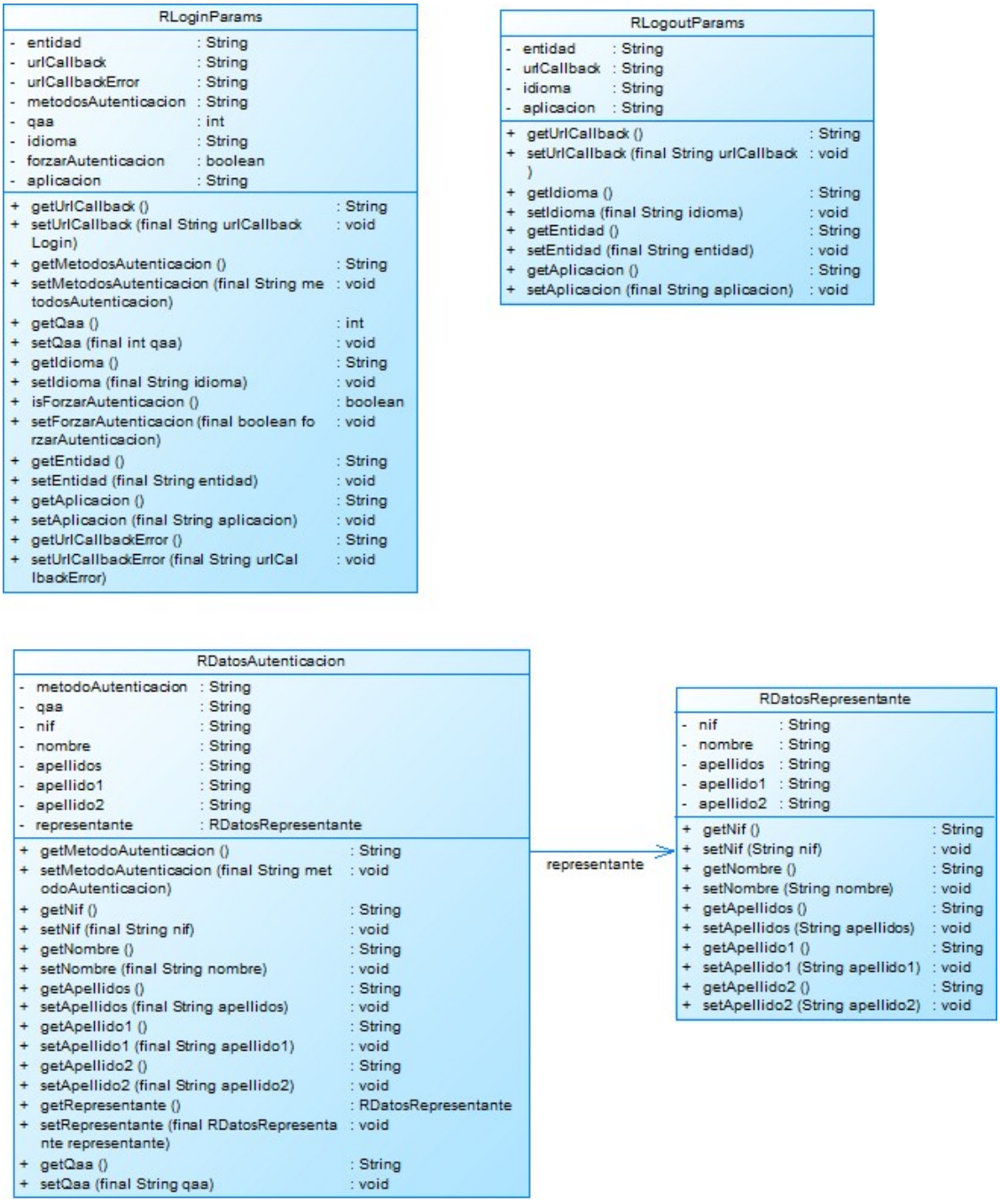
- urlCallbackLogin: URL a la que s'ha de redirigir tras finalitzar el procés d'autenticació. A aquesta URL se li passarà per POST un paràmetre amb un ticket per obtenir la informació d'autenticació.
- urlCallbackError: URL a la que s'ha de redirigir si es produeix un error al procés d'autenticació.
- mètodes: mètodes d'autenticació suportats (ànonim, aFirma, AEAT y SS)
- idioma: idioma

LoginIB genera sessió d'autenticació i com a resposta a la petició retorna URL a la que l'aplicació origen ha de redirigir el navegador per mostrar la pàgina de login.

2. L'aplicació ha de redirigir el navegador a la URL retornada.
3. En funció del mètode d'autenticació seleccionat per l'usuari a la pàgina de login, LoginIB durà a terme l'operativa corresponent. En el cas de Cl@ve, LoginIB prepara la petició d'autenticació a Cl@ve i la signa. En aquesta petició s'indica la URL de retorn que ha d'emprar Cl@ve per retornar a LoginIB. Se redirigeix el navegador de l'usuari a Cl@ve passant per POST la petició d'autenticació. En cas d'accés anònim s'obviaran les passes 4 i 5, que es descriuen a continuació, i es passarà directament al 6.
4. Cl@ve procedeix a autenticar l'usuari segons el mètode seleccionat per aquest.
5. Cl@ve redirigeix el navegador a la URL de retorn passant per POST un paràmetre amb la resposta signada, que conté les dades de l'autenticació. LoginIB processa la resposta la resposta de Cl@ve i amb les dades d'autenticació s'actualitza el tiquet.
6. LoginIB redirigeix a la URL de callback de login passant el tiquet.
7. L'aplicació consumidora invoca el servei de LoginIB per obtenir la informació d'autenticació a partir del tiquet. Una vegada emprat el tiquet, aquest s'esborra i no es pot tornar a emprar. Se procedeix amb l'autenticació i es valida l'usuari a l'aplicació origen.
8. Una vegada validat l'usuari, es permet l'accés al recurs.

3. Model de classes

A continuació es presenta el model de classes de l'API REST de LoginIB.



Donat que aquesta API està documentada mitjançant Swagger, es pot consultar el detall del camps d'aquestes classes a la URL http://host:port/loginibws/rest, on *host* i *port* hauràn de ser substituïts per les dades de l'entorn emprat.

4. Descripció dels serveis

Tota la informació relativa al serveis de l'API, paràmetres d'entrada i valors de retorn es pot trobar a la pàgina de Swagger (http://host:port/loginibws/rest).

5. Consideracions de disseny i ús de l'API

5.1. *Endpoints* dels serveis

A continuació s'indiquen els *endpoints* dels diferents serveis de l'API de LoginIB. S'haura de revisar el valor de «host» i «port» en funció de l'entorn al que s'hi vulgui accedir.

Servei d'inici de sessió d'autenticació:

```
http://host:port/loginibws/rest/v1/login
```

Servei de recuperació de dades d'autenticació a partir de tiquet:

```
http://host:port/loginibws/rest/v1/ticket/{ticket}
```

Servei de fi de sessió d'autenticació:

```
http://host:port/loginibws/rest/v1/logout
```

5.2. Securització dels serveis

Els serveis de l'aplicació estan securitzats amb autenticació BASIC amb usuari/contrasenya.

L'usuari emprar per a consumir els serveis haurà de tenir assignat el rol LIB_API.

5.3. Dependències Maven

Per a desenvolupar un client per al servei d'autenticació el més senzill és crear un nou projecte amb Maven i incloent els següents blocs a dins el fitxer pom.xml:

A dins la secció <dependencies>:

```
<dependency>
	<groupId>es.caib.loginib</groupId>
	<artifactId>loginib-rest-api</artifactId>
	<version>0.0.1-SNAPSHOT</version>
</dependency>
<dependency>
	<groupId>org.springframework</groupId>
	<artifactId>spring-web</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<scope>provided</scope>
</dependency>
```

A dins la secció <repositories>:

```
        <repository>
              <id>github-governib-maven-repo</id>
              <name>GitHub GovernIB Maven Repository</name>
              <url>http://GovernIB.github.io/maven/maven/</url>
        </repository>
```

6. Client d'exemple

No existeix un client de referencia pel consum dels serveis de l'API de LoginIB, però a continuació es presenta, a mode d'exemple, una classe test que consumeix els serveis emprant Spring Web.

```
public class TestLoginIB {

	private static final String METODES_AUTENTICACIO =
"ANONIMO;CLAVE_CERTIFICADO;CLAVE_PIN;CLAVE_PERMANENTE";
	private static final String CODI_ENTITAT = "A04003003";
	private static final String APLICACIO_CODI = "CLIENT_TEST";
	private static final String URL_CALLBACK_LOGIN = "http://host:port/recurs";
	private static final String URL_CALLBACK_ERROR = "http://host:port/error";
	private static final String URL_CALLBACK_LOGOUT = "http://host:port/logout";
	private static final String IDIOMA = "ca";
	private static final String NIVELL_QAA = "3";
	private static final String USUARI = "usu";
	private static final String CONTRASENYA = "pass";
	private static final String URL = "http://host:port/loginibws/rest/v1";

	private String iniciarSesionAutenticacion(){

	  final RLoginParams param = new RLoginParams();
	  param.setAplicacion(APLICACIO_CODI);
	  param.setEntidad(CODI_ENTITAT);
	  param.setUrlCallback(URL_CALLBACK_LOGIN);
	  param.setUrlCallbackError(URL_CALLBACK_ERROR);
	  param.setIdioma(IDIOMA);
	  param.setForzarAutenticacion(false);
	  param.setQaa(Integer.parseInt(NIVELL_QAA));
	  param.setMetodosAutenticacion(METODES_AUTENTICACIO);
	  final RestTemplate restTemplate = new RestTemplate();

	  restTemplate.getInterceptors().add(new BasicAuthorizationInterceptor(
	          USUARI, CONTRASENYA));

	  final HttpHeaders headers = new HttpHeaders();
	  headers.setContentType(MediaType.APPLICATION_JSON);

	  final HttpEntity<RLoginParams> request = new HttpEntity<>(param,
	          headers);
	  final ResponseEntity<String> responseTramite = restTemplate
	          .postForEntity(URL + "/login", request,
	                  String.class);

	  return responseTramite.getBody();
	}

	private RDatosAutenticacion validarTicketAutenticacion(final String pTicket){

	  final RestTemplate restTemplate = new RestTemplate();
```

```
        restTemplate.getInterceptors().add(new BasicAuthorizationInterceptor(
                USUARI, CONTRASENYA));

        final RDatosAutenticacion datosAutenticacion = restTemplate.getForObject(
                URL + "/ticket/" + pTicket,
                RdatosAutenticacion.class);

        return datos;
    }

    private String iniciarSesionLogout(){

        final RestTemplate restTemplate = new RestTemplate();

        restTemplate.getInterceptors().add(new BasicAuthorizationInterceptor(
                USUARI, CONTRASENYA));

        final RLogoutParams param = new RlogoutParams();
        param.setAplicacion(APLICACIO_CODI);
        param.setEntidad(codigoEntidad);
        param.setUrlCallback(URL_CALLBACK_LOGOUT);
        param.setIdioma(IDIOMA);
        final HttpHeaders headers = new HttpHeaders();
        headers.setContentType(MediaType.APPLICATION_JSON);

        final HttpEntity<RLogoutParams> request = new HttpEntity<>(param,
                headers);
        final ResponseEntity<String> responseTramite = restTemplate
                .postForEntity(URL + "/logout", request,
                        String.class);

        return responseTramite.getBody();
     }
}
```