# Liberación SISTRA: módulo Login

Código: DSI-LIBSISTRA-LOGIN

# HOJA DE CONTROL DE DOCUMENTO

## DOCUMENTO/ARCHIVO

| | |
|---|---|
| Título: Liberación SISTRA: módulo Login | Nombre Archivo/s: SISTRA-PLUGINLOGIN-002.doc |
| Código: SISTRA-PLUGINLOGIN | Soporte lógico: Word |
| Fecha: septiembre 2010 | |
| Versión: 2 | |

## REGISTRO DE CAMBIOS

| Versión | Páginas | Motivo del cambio |
|---|---|---|
| 1 | 8 | Creación del Documento |
| 2 | 12 | Integración login CAS |
| | | |

## DISTRIBUCIÓN DEL DOCUMENTO

| Nombre | Personal |
|---|---|
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |

## CONTROL DEL DOCUMENTO

| PREPARADO | REVISADO/ APROBADO | ACEPTADO | ACEPTADO |
|---|---|---|---|
| Rafael Sanz | José Vicente Juan Pérez | **Jeroni Navarrete** | **Bernat Alberti** |

***Cumplimentar con el nombre, la firma y la fecha*** ***Sólo para clientes***

# INDICE

# 1. Objeto

El objeto de este documento es definir las acciones a realizar para permitir el uso de la plataforma SISTRA sin el módulo de login personalizado de la DGTIC.

# 2. Módulo de login DGTIC

La DGTIC ha definido un módulo de login personalizado fuertemente ligado a su repositorio de usuarios SEYCON que tiene las siguientes características:

- Single Sign On corporativo dentro de la CAIB (acoplado a la gestión de usuarios de la DGTIC)
- Permite 3 modos de acceso: anónimo, autenticado por usuario/password y autenticado por certificado
- Utiliza el módulo de firma de la DGTIC
- Desarrollado para uso exclusivo en el jboss customizado de la CAIB

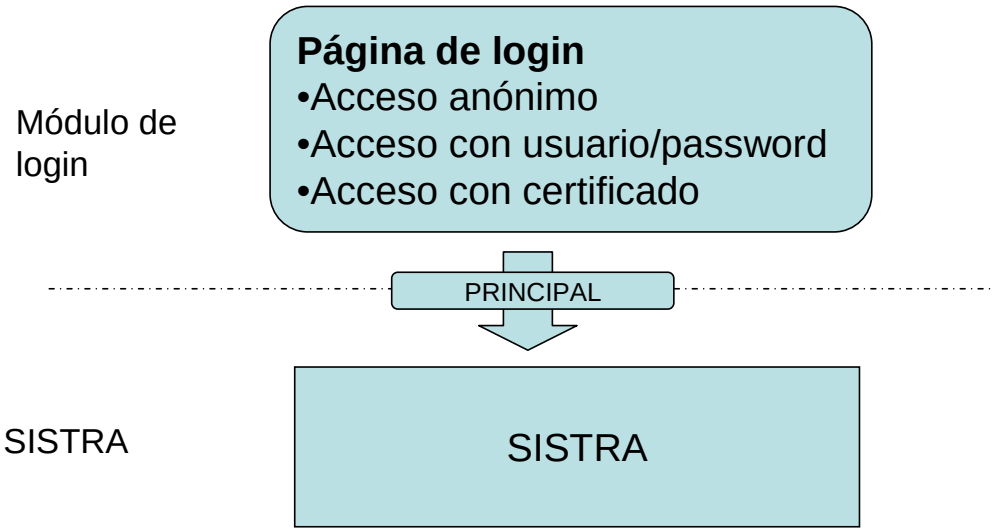
# 3. Funcionalidades requeridas del módulo de login

La plataforma de SISTRA esta basada en los estándares de seguridad de J2EE, según los cuales una aplicación espera recibir una identidad (Principal) que tenga definidos unos permisos de acceso (Roles).

Desde el punto de vista de la plataforma el único requisito es que se debe recibir una identidad (Principal) que nos indique las siguientes propiedades:

- método de acceso (anónimo, autenticado por usuario/password o autenticado por certificado)
- nif/cif del usuario autenticado (en caso de que no sea anónimo)
- nombre del usuario autenticado (en caso de que no sea anónimo)

El acceso anónimo será un acceso autenticado con usuario especial (nobody).



# 4. Solución propuesta

Para que la plataforma SISTRA se libere de forma que no comprometa el uso del módulo de login de la DGTIC (ni ninguna otra solución de Single Sign On) se propone liberar la plataforma con una solución que haga la plataforma operativa pero que no obligue a adoptar ninguna solución de Single Sign On.

Es tarea del organismo decidir cuál será su solución más adecuada para la gestión corporativa del Single Sign On y estudiar cómo se debe integrar con la plataforma SISTRA.

La solución de Single Sign On con la que se liberará SISTRA y que servirá como ejemplo de la funcionalidad requerida se basará en:

- implementación de un loginmodule JAAS para Jboss. Este loginmodule funcionará contra unas tablas de usuarios de test. Se proveerán los fuentes de este loginmodule por si se quiere modificar por el organismo para hacerlo funcionar contra otras tablas, ldap, etc.
- páginas de login ubicadas en cada módulo web que implementarán un acceso similar al que ofrece la página de login de la DGTIC. Para el acceso con certificados se utilizará el plugin de @firma, ya que se supone que será la DGTIC la única que utilice su api de firma.
- para la gestión del Single Sign On se utilizará la gestión incorporada por JBoss. Jboss tiene embebido un Apache Tomcat el cual proporciona un mecanismo de SSO usando una valve (http://www.jboss.org/community/docs/DOC-12280)

# 5. Interfaz Principal

El interfaz que debe implementar el Principal autenticado es la siguiente:

| Método | Descripción | Parámetros | Resultado |
|---|---|---|---|
| getMetodoAutenticacion | Obtiene método de autenticación | Principal: principal autenticado | Método autenticación |
| getNif | Devuelve el nif/cif del usuario autenticado (en caso de que no sea anónimo) | Principal: principal autenticado | Nif |
| getNombreCompleto | Obtiene el nombre completo del usuario autenticado (en caso de que no sea anónimo) | Principal: principal autenticado | Nombre completo |

*Para más información consultar el javadoc*

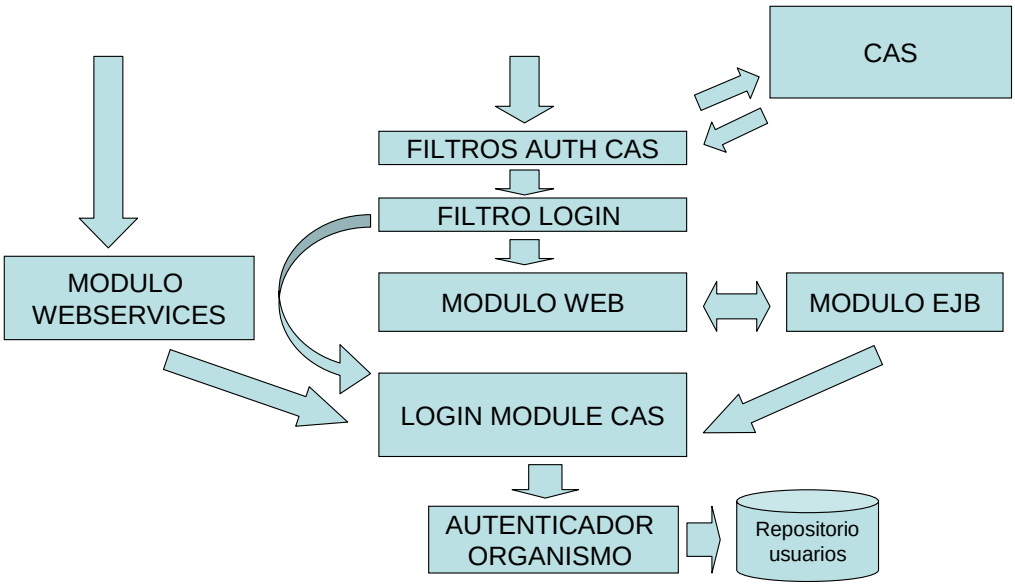
# 6. ANEXO I: Integración con CAS

Sistra ofrece un plugin de login que permite integrarse con el servicio de single sign on CAS (Central Authentication Service) de Jasig.

CAS está basado en filtros web específicos que implementan la autenticación personalizada de CAS. Esta autenticación no se traslada al contenedor web (funciones getPrincipal y isUserInRole de la HttpServletRequest), sino que son simulados a través de un wrapper de la request que intercepta la request original. Al no trasladarse al contenedor web no se propaga la información de seguridad al contenedor de EJB al llamar de la capa web a la capa EJB.

Para realizar está integración se ha diseñado la siguiente solución:

- inclusión de los filtros web de autenticación de CAS en todos los módulos web
- inclusión de un filtro web personalizado que realiza el login en el contenedor web, para que la información se traslade al contenedor de EJB.
- Login module que recibe el principal validado por CAS en la capa web y lo autentica. Este login module puede recibir dos tipos de peticiones:
    - referencia a usuarios validados por CAS, para los cuales se fía de su autenticación
    - usuario / password (p.e. para capa de webservices) para el que lo autenticará contra la fuente de datos de usuarios. Para poder personalizar la fuente de datos de usuarios se ha externalizado esta autenticación en una clase que deberá ser implementada según la implantación en el organismo.




Por otra parte CAS deberá ser particularizado para que Sistra pueda recibir la siguiente información en los atributos del usuario autenticado por CAS:

- Nif: Nif del usuario autenticado
- Nombre: Nombre completo del usuario autenticado
- Método de autenticación: C (Certificado) / U (Usuario) / A (Anónimo)

Para realizar la integración de Sistra con CAS hay que realizar los siguientes pasos:

- Compilar el proyecto indicando que que se trabaja con CAS:
  - En el fichero config.properties que establece las opciones de compilación del proyecto establecer las siguientes propiedades:

```
# Configuracion login: JAAS o CAS
login-config=CAS


# Login CAS: info necesaria para los filtros de CAS
#   - Url donde esta CAS
cas.urlCas=https://hostnameCAS/cas
#     - Url donde esta sistra
cas.urlSistra=https://hostnameSistra
#     - Url donde esta sistra (separada para fronts (sistrafront,formfront,zonaperfront,redosefront) y backs (resto)
cas.urlSistra.front=https://hostnameSistraFronts
cas.urlSistra.back=https://hostnameSistraBacks
```

- Compilar el módulo de integración con Sistra: /integracio/llibreria-casClient a partir del build.xml ubicado en dicho directorio. Se generará el jar: /integracio/llibreria-casClient/output/product/sistra-casClient.jar

- Incluir las siguientes librerías en el default/lib de Jboss:

  cas-client-core-3.1.10.jar

  saaj-api.jar

  saaj-impl.jar

  loginModuleCIM.jar

  opensaml-1.1b.jar

  sistra-casClient.jar

  xmlsec-1.3.0.jar

- Eliminar del default/lib de JBoss la implementación de saaj de JBoss ya que es demasiado antigua para usar con CAS:

  jboss-saaj.jar

- Asegurarse de que se ha actualizado el stack de xml de JBoss. Para ello copiar a /lib/endorsed las librerías:

  resolver.jar

  xercesImpl.jar

  xml-apis.jar

- Modificar la configuración de login de JBoss para añadir el login module para CAS:

```
<application-policy name = "seycon">
```

```
 <authentication>
<login-module code = "es.caib.sistra.casClient.loginModule.CasInternalLoginModule"
        flag = "sufficient">
                <module-option name="unauthenticatedIdentity">nobody</module-option>
 </login-module>
 </authentication>
</application-policy>
```

- Por último habrá que establecer el fichero de propiedades del plugin de CAS e indicar la clase que realiza la autenticación de usuarios para usuarios no provenientes de CAS. Este fichero de propiedades debe ir ubicado en <dir_config> \sistra\plugins\ plugin-login.properties

```
# Atributos donde se recogen los datos del usuario autenticado
cas.nifAttribute=nif
cas.nombreAttribute=nombreApellidos
cas.metodoAutenticacionAttribute=metodoAutenticacion
# Roles: en caso de que se recuperen de un atributo de la información
# recogida por CAS se indicará el nombre de la propiedad. Si los roles
# los establece la clase autenticadora propia del organismo se establecerá
# como INTERNAL
cas.rolesAttribute=INTERNAL
#cas.rolesAttribute=roles

# Clase autenticadora
cas.loginModule.autenticador=es.xxx.xxx.xxx
```

La clase autenticadora a implementar por el organismo debe cumplir la siguiente interface es.caib.sistra.casClient.loginModule.AutenticadorInt:

```
/**
 * Autentica el usuario contra la fuente de usuarios
 * @param propsPlugin Propiedades del plugin
 * @param user Username
 * @param pass Password
 * @return En caso correcto devuelve la información del usuario. En caso incorrecto devuelve nulo.
 */
public UserInfo autenticar(Properties propsPlugin, String user,String pass);

/**
 * Obtener lista de roles de un usuario. Se usará cuando la lista de roles no sea proporcionada por CAS

 * @param propsPlugin Propiedades del plugin
 * @param user Username
 * @return En caso correcto devuelve lista de roles del usuario. En caso incorrecto devuelve nulo.
```

```
*/
public List<String> obtenerRoles(Properties propsPlugin, String user);
```