Integración de Pentaho BI Server con Jasig CAS

Guía de Instalación Rápida

Stratebi Business Solutions. (2016)

www.stratebi.com

Índice

1 Introducción

Esta guía rápida de instalación, tiene como objetivo desplegar en un entorno local el Servidor Analítico de Pentaho *(**P**entaho **B**usiness **I**ntelligence **S**erver, **BISERVER**)* en su versión **5.4.0.1 Edición Comunitaria**, configurado para que haga uso del protocolo del ***S**ervicio de **A**utenticación **C**entralizado (**C**entral **A**uthentication **S**ervice, **CAS**)*.

CAS permite el acceso a un usuario a múltiples sistemas con una único proceso de identificación *(**S**ingle **si**gn-**o**n, **SSO**)* y requiere que todas las comunicaciones establecidas se realicen a través de la Capa de Conexión Segura *(**S**ecure **S**ockets **L**ayer, **SSL**), por lo cual,* un certificado **X.509** es necesario para este despliegue.

2 Licencia

3 Requisitos

3.1 Software requerido en el despliegue

Software	Versión	Tamaño	Descarga
Pentaho Business Intelligence Server (BISERVER)	5.4.0.1-130	814,8 MB	biserver-ce-5.4.0.1-130.zip https://sourceforge.net/projects/pentaho/files/Business%20Intelligence%20Server/5.4/biserver-ce-5.4.0.1-130.zip/download
Apache Tomcat	6.0.45	7,0 MB	apache-tomcat-6.0.45.zip http://ftp.cixug.es/apache/tomcat/tomcat-6/v6.0.45/bin/apache-tomcat-6.0.45.zip
Jasig Central Authentication Service (JCAS)	3.5.3	32,3 MB	cas-server-webapp-3.5.3.war http://central.maven.org/maven2/org/jasig/cas/cas-server-webapp/3.5.3/cas-server-webapp-3.5.3.war
Jasig CAS Client (JCASC) For Java Core	3.1.12	85 KB	cas-client-core-3.1.12.jar http://central.maven.org/maven2/org/jasig/cas/client/cas-client-core/3.1.12/cas-client-core-3.1.12.jar
Spring Security CAS Support (SSCASS)	2.0.8	16 KB	spring-security-cas-client-2.0.8.RELEASE.jar http://central.maven.org/maven2/org/springframework/security/spring-security-cas-client/2.0.8.RELEASE/spring-security-cas-client-2.0.8.RELEASE.jar
OpenJDK	1.7	-	Depende del sistema operativo a utilizar.

3.2 Entorno requerido en el despliegue

Hardware - 64 bit	Sistema Operativo - 64 bit
Procesador • Apple Macintosh Pro de cuatro núcleos o Macintosh Mini de cuatro núcleos. • Intel EM64T o AMD64 de doble núcleo. **RAM** • 8 GB con 4 GB dedicadas exclusivamente al BISERVER. **Disco** • 10 GB para la instalación.	• Apple Macintosh OS X Server 10.9 & 10.10. • CentOS Linux 5 & 6. • Microsoft Windows 2008 Server R2, 2012 Server R2. • Red Hat Enterprise Linux 5 & 6. • Solaris 10. • Ubuntu Server 12.04 LTS & 14.04 LTS.

4 Descripción del entorno

La presente guía asume que el entorno a utilizar posee las siguientes características:

- El usuario con el que se realiza todo el despliegue **no posee permisos de administrador** y su nombre de **stratebi**.
- El nombre del equipo es **pentaho**.
- La estructura de los directorios a utilizar es la siguiente:
  1 **BASE**. Directorio raíz. Puede tener cualquier nombre.
    1.1 **APP**. Destino final de los ficheros del despliegue.
    1.1 **ARCHIVE**. Directorio para respaldo de los ficheros desplegados en **APP**.
    1.1 **TMP**. Directorio para ficheros intermedios o despliegues de prueba.
    1.2 **SOURCE**. Directorio donde residen todos los ficheros base utilizados durante el despliegue.
- Cualquier comando descrito utiliza rutas relativas tomando el directorio **BASE** como punto de partida.
- Toda referencia a ficheros, es descrita con rutas relativas, utilizando el directorio **BASE** como punto de partida.
- Los puertos necesarios para el correcto funcionamiento de los servidores desplegados han de encontrarse disponibles y accesibles. Esta guía garantiza que ambos servidores instalados no tendrán conflicto de puerto.

5 Instalación

De forma general, los pasos a seguir son los siguientes:

1. Crear un certificado auto firmado para cifrar las comunicaciones entre el BISERVER y el CAS. Este certificado será utilizado por ambos servidores.
2. Instalar Apache Tomcat modificando sus puertos predeterminando para que no coincidan con los puertos utilizados por el BISERVER ya que este también es desplegado dentro de un Apache Tomcat. Además, definimos un nuevo **Connector** que será el encargado de recibir las peticiones cifradas utilizando el certificado creado en el primer paso.
3. Installar el BISERVER.
4. Installar los componentes necesarios para soporte de CAS en el BISERVER.

Note que los puntos 1, 2 y 3 se utilizan para el despliegue del CAS en un entorno local, por lo que puede prescindir de ellos si ya posee un CAS desplegado y funcionando.

5.1 Creación de certificado X.509

La implementación de CAS utilizada en esta guía, funciona únicamente si logra establecer una conexión segura con cliente que realiza la solicitud, por lo tanto, un certificado **X.509** es requerido para habilitar este tipo de conexiones en Apache Tomcat. Dicho esto, crearemos un certificado auto firmado que será almacenado dentro de un fichero de tipo ***J**ava **K**ey**S**tore (**JKS**)* con nombre **cas-certificate.jks**, que no es más que un contenedor cifrado utilizado para guardar certificados. Dicho contenedor será utilizado tanto por el Servidor CAS como por el BISERVER.

5.1.1 Requisitos

- Java Runtime Environment o Java Development Kit para la creación del certificado auto firmado.

5.1.2 Pasos de creación

1. Cree el certificado y su contenedor utilizando la herramienta **keytool** provista por Java.

```
[stratebi@pentaho base]$                                                    Terminal
[stratebi@pentaho base]$ keytool -genkeypair -alias cas -keyalg RSA -dname "CN=localhost,OU=Quality &
Assurance,O=Stratebi,L=Madrid,S=MA,C=ES" -keypass password -keystore cas-certificate.jks -storepass
password
[invitado@CentOS casguide]$
```

2. Copie el fichero **cas-certificate.jks** en el directorio **base > app,** donde serán instalados los

servidores.

5.2 Instalación del Servidor CAS

***J**asig **C**entral **A**uthentication **S**ervice (**JCAS**)*, es una implementación del protocolo CAS desarrollado por la comunidad de Apereo. Se distribuye como aplicación web en un fichero ***WAR** (**W**eb **A**pplication **A**rchive),* por lo cual, utilizaremos Apache Tomcat como servidor web para su instalación.

5.2.1 Requisitos

- Java Runtime Environment o Java Development Kit para ejecutar del servidor Apache Tomcat.
- Ubicación del contenedor con el certificado a utilizar para la configuración de la conexión segura hacia el Servidor CAS donde residirá el JCAS.

5.2.2 Pasos de instalación

1. Descargue Apache Tomcat en la carpeta **source**.
2. Descomprima el fichero **source > apache-tomcat-{version}.zip** en la carpeta de **app**. Esto crea un directorio nuevo con el nombre de **apache-tomcat-{version}**.
3. Abra el fichero de configuración del servidor de Apache Tomcat, **app > apache-tomcat-{version} > conf > server.xml** para cambiar los puertos de conexión por defecto. Esto es necesario para que los puertos del servidor CAS no coincida con los del BISERVER, ya que este último se despliega dentro de su propio Tomcat.
   (a) Modifique el atributo **port** del elemento **Server**, de 8005 a **8006** (línea 22 apróx.).

```
22. <Server port="8006" shutdown="SHUTDOWN">                                    Texto
23.   <Listener className="org.apache.catalina.startup.VersionLoggerListener" />
```

   (b) Modifique el atributo **port** y **redirectPort** del elemento **Connector** de 8080 a **8088** y de 8443 a **8444** respectivamente (entre líneas 71 - 73 apróx.).

```
71. <Connector port="8088" protocol="HTTP/1.1"                                  Texto
72.            connectionTimeout="20000"
73.            redirectPort="8444" />
```

   (c) Configure la conexión segura, removiendo las etiquetas **<!-- -->** que rodean al elemento **Connector** (entre líneas 83 - 87 apróx.). Luego, modifique el atributo **port** de la etiqueta **Connector**, de 8443 a **8444** y agregue los siguiente atributos: **keystoreFile="../cas-certificate.jks" keystorePass="password" keyAlias="cas"**. Note que la ruta utilizada en el atributo keystoreFile es relativa al directorio donde se ha desplegado el Apache Tomcat.

```
83.                                                                             Texto
84. <Connector port="8444" protocol="HTTP/1.1" SSLEnabled="true"
85.            keystoreFile="../cas-certificate.jks" keystorePass="password" keyAlias="cas"
86.            maxThreads="150" scheme="https" secure="true"
87.            clientAuth="false" sslProtocol="TLS" />
```

   (d) Modifique el atributo **port** y **redirectPort** del elemento **Connector** de 8009 a **8010** y de 8443 a **8444** respectivamente para modificar el puerto de conexión hacia el servidor a través del protocolo AJP (línea 90 apróx.).

```
89. <!-- Define an AJP 1.3 Connector on port 8009 -->                           Texto
90. <Connector port="8010" protocol="AJP/1.3" redirectPort="8444" />
```

4. Guarde los cambios realizados y cierre el documento.
5. Descargue JCAS, **cas-server-webapp-{versión}.war**, en el directorio **app > apache-tomcat-{version} > webapps** del Apache Tomcat. Al momento de ejecutar el servidor Apache Tomcat, este descomprime el fichero **cas-server-webapp-{versión}.war** en el mismo directorio donde fue colocado con el nombre de **cas-server-webapp-{versión}**, por lo cual, no es necesario realizar pasos adicionales.

5.3 Instalación del BISERVER

5.3.1 Requisitos

- Java Runtime Environment o Java Development Kit para ejecutar el BISERVER.
- Servidor CAS configurado y funcionando con certificado.
- Ubicación del contenedor del certificados de confianza que contenga el certificado del Servidor CAS.

5.3.2 Pasos de instalación

1. Descargue el BISERVER en la carpeta **source**.
2. Descomprima el fichero **source > biserver-ce-5.4.0.1-130.zip** en el directorio **app.** Este creará un nuevo directorio con nombre **biserver-ce**.
3. Cambie el nombre de directorio recién creado a **biserver-ce-5.4.0.1**. Esto sirve para recordar que versión de BISERVER tenemos instalada.
4. Opcionalmente, genere un enlace entre directorios para hacer referencia a la última versión de BISERVER en uso.

```
[stratebi@pentaho base]$ unzip ./source/biserver-ce-5.4.0.1-130.zip -d ./app/
[stratebi@pentaho base]$ mv ./app/biserver-ce/ ./app/biserver-ce-5.4.0.1
[stratebi@pentaho base]$ ln -s ./biserver-ce-5.4.0.1 app/biserver
[stratebi@pentaho base]$
```
Terminal

5. Duplique el fichero **app > biserver-ce-5.4.0.1 > start-pentaho.sh** y renombre la copia a **start-pentaho-ssl-trust.sh**, esta última servirá para definir la ruta del contenedor de certificados de confianza que el BISERVER ha de utilizar al conectarse al Servidor CAS. Para ello, modifique el fichero **app > biserver-ce-5.4.0.1 > start-pentaho-ssl-trust.sh**, e inserte dos nuevas variables de entorno para la máquina virtual de java: **javax.net.ssl.trustStore**, que almacena la ruta del fichero donde se encuentra el contenedor de confianza del certificado y **javax.net.ssl.trustStorePassword** que almacena la contraseña para poder acceder al certificado almacenado dentro del contenedor previamente definido (línea 24 aprox.).

```
21. ...
22. if [ "$errCode" = 0 ]; then
23.   cd "$DIR/tomcat/bin"
24.   CATALINA_OPTS="-Xms1024m -Xmx2048m -XX:MaxPermSize=256m
          -Dsun.rmi.dgc.client.gcInterval=3600000 -Dsun.rmi.dgc.server.gcInterval=3600000
          -Djavax.net.ssl.trustStorePassword=password
          -Djavax.net.ssl.trustStore=/home/stratebi/base/app/cas-certificate.jks"
25.   export CATALINA_OPTS
26.   JAVA_HOME=$_PENTAHO_JAVA_HOME
27.   sh startup.sh
28. fi
```
Texto

5.4 Instalación del soporte para CAS en BISERVER

5.4.1 Requisitos

- BISERVER previamente instalado.
- Dirección web del Servidor CAS: `https://localhost:8444/cas-server-webapp-3.5.3/`.

5.4.2 Pasos de instalación

1. Descargue **Jasig CAS Client** y **Spring Security CAS Support** en la carpeta **source**.
2. Copie los ficheros **source > spring-security-cas-client-2.0.8.RELEASE.jar** y **source > cas-client-core-3.1.12.jar**, en el directorio **app > biserver-ce-5.4.0.1 > tomcat > webapps > pentaho > WEB-INF > lib** junto a las otras librerías Java.

```
[stratebi@pentaho base]$
[stratebi@pentaho base]$ cp ./source/spring-security-cas-client-2.0.8.RELEASE.jar ./source/cas-client-core-3.1.12.jar app/biserver-ce-5.4.0.1/tomcat/webapps/pentaho/WEB-INF/lib
[stratebi@pentaho base]$
```
Terminal

3. Cree el fichero **app > biserver-ce-5.4.0.1 > pentaho-solutions > system> applicationContext-**

spring-security-cas.xml e incluya en él, el contenido que se muestra a continuación. Preste particular atención a las líneas resaltadas:

Texto

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:pen="http://www.pentaho.com/schema/pentaho-system"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
       http://www.pentaho.com/schema/pentaho-system http://www.pentaho.com/schema/pentaho-system.xsd" default-lazy-
init="true">

  <bean id="serviceProperties" class="org.springframework.security.ui.cas.ServiceProperties">
    <property name="service" value="http://localhost:8080/pentaho/j_spring_cas_security_check" />
    <property name="sendRenew" value="false" />
  </bean>

  <bean id="casProcessingFilter" class="org.springframework.security.ui.cas.CasProcessingFilter">
    <property name="authenticationManager" ref="authenticationManager" />
    <property name="authenticationFailureUrl" value="https://localhost:8444/cas-server-webapp-
3.5.3/authorizationFailure.jsp" />
    <property name="defaultTargetUrl" value="/" />
    <property name="filterProcessesUrl" value="/j_spring_cas_security_check" />
  </bean>

  <bean id="casProcessingFilterEntryPoint" class="org.springframework.security.ui.cas.CasProcessingFilterEntryPoint">
    <property name="loginUrl" value="https://localhost:8444/cas-server-webapp-3.5.3/login"/>
    <property name="serviceProperties" ref="serviceProperties" />
  </bean>

  <bean id="casAuthenticationProvider" class="org.springframework.security.providers.cas.CasAuthenticationProvider">
    <property name="userDetailsService">
      <pen:bean class="org.springframework.security.userdetails.UserDetailsService"/>
    </property>
    <property name="serviceProperties" ref="serviceProperties"/>
    <property name="ticketValidator">
      <bean class="org.jasig.cas.client.validation.Cas20ServiceTicketValidator">
        <constructor-arg index="0" value="https://localhost:8444/cas-server-webapp-3.5.3" />
      </bean>
    </property>
    <property name="key" value="an_id_for_this_auth_provider_only"/>
    <pen:publish as-type="org.springframework.security.providers.AuthenticationProvider">
      <pen:attributes>
        <pen:attr key="providerName" value="cas"/>
      </pen:attributes>
    </pen:publish>
  </bean>

   <bean id="filterChainProxy" class="org.springframework.security.util.FilterChainProxy">
    <property name="filterInvocationDefinitionSource">
      <value>
        <![CDATA[CONVERT_URL_TO_LOWERCASE_BEFORE_COMPARISON
        PATTERN_TYPE_APACHE_ANT

/webservices/**=securityContextHolderAwareRequestFilterForWS,httpSessionPentahoSessionContextIntegrationFilter,httpSess
ionContextIntegrationFilter,casProcessingFilter,basicProcessingFilter,anonymousProcessingFilter,exceptionTranslationFil
terForWS,filterInvocationInterceptorForWS

/api/**=securityContextHolderAwareRequestFilterForWS,httpSessionPentahoSessionContextIntegrationFilter,httpSessionConte
xtIntegrationFilter,casProcessingFilter,basicProcessingFilter,anonymousProcessingFilter,exceptionTranslationFilterForWS
,filterInvocationInterceptorForWS

/plugin/**=securityContextHolderAwareRequestFilterForWS,httpSessionPentahoSessionContextIntegrationFilter,httpSessionCo
ntextIntegrationFilter,casProcessingFilter,basicProcessingFilter,anonymousProcessingFilter,exceptionTranslationFilterFo
rWS,filterInvocationInterceptorForWS

/**=securityContextHolderAwareRequestFilter,httpSessionPentahoSessionContextIntegrationFilter,httpSessionContextIntegra
tionFilter,httpSessionReuseDetectionFilter,logoutFilter,casProcessingFilter,authenticationProcessingFilter,basicProcess
ingFilter,requestParameterProcessingFilter,anonymousProcessingFilter,exceptionTranslationFilter,filterInvocationInterce
ptor]]>
      </value>
    </property>
  </bean>

  <bean id="authenticationManager" class="org.springframework.security.providers.ProviderManager">
    <property name="providers">
      <list>
        <ref bean="casAuthenticationProvider" />
        <ref bean="anonymousAuthenticationProvider" />
      </list>
    </property>
  </bean>

  <bean id="exceptionTranslationFilter" class="org.springframework.security.ui.ExceptionTranslationFilter">
```

```
    <property name="authenticationEntryPoint">
      <ref local="casProcessingFilterEntryPoint" />
    </property>
    <property name="accessDeniedHandler">
      <bean class="org.springframework.security.ui.AccessDeniedHandlerImpl" />
    </property>
  </bean>

  <bean id="exceptionTranslationFilterForWS" class="org.springframework.security.ui.ExceptionTranslationFilter">
    <property name="authenticationEntryPoint">
      <ref local="casProcessingFilterEntryPoint" />
    </property>
    <property name="accessDeniedHandler">
      <bean class="org.springframework.security.ui.AccessDeniedHandlerImpl" />
    </property>
  </bean>

  <bean id="logoutFilter" class="org.springframework.security.ui.logout.LogoutFilter">
    <constructor-arg value="https://localhost:8444/cas-server-webapp-3.5.3/logout?
url=http://localhost:8080/pentaho/Home" />
    <!-- URL redirected to after logout -->
    <constructor-arg>
      <list>
        <bean class="org.pentaho.platform.web.http.security.PentahoLogoutHandler" />
        <bean
            class="org.springframework.security.ui.logout.SecurityContextLogoutHandler" />
      </list>
    </constructor-arg>
    <property name="filterProcessesUrl" value="/Logout" />
  </bean>
</beans>
```

4. Modifique el fichero **app > biserver-ce-5.4.0.1 > pentaho-solutions > system > pentaho-spring-beans.xml**, agregando una entrada nueva inmediatamente después de **<import resource="applicationContext-spring-security-jdbc.xml" />** con el siguiente contenido (línea 91 apróx.): **<import resource="applicationContext-spring-security-cas.xml" />**.

```
89. ...
90. <import resource="applicationContext-pentaho-security-jdbc.xml" />
91. <import resource="applicationContext-spring-security-jdbc.xml" />
92. <import resource="applicationContext-spring-security-cas.xml" />
93.
```
Texto

5.5 Pruebas

Una ves terminada la instalación, debemos levantar ambos servidores y comprobar que el acceso al BISERVER sea satisfactorio. Para ello debemos:

1. Arranque el Servidor CAS.
   (a) Vaya al directorio **app > apache-tomcat-{version} > bin** y inicie el servidor utilizando el script **startup.sh**.

```
[stratebi@pentaho base]$
[stratebi@pentaho base]$ cd ./app/apache-tomcat-6.0.45/bin
[stratebi@pentaho bin]$ ./startup.sh
Using CATALINA_BASE:   /home/stratebi/base/app/apache-tomcat-6.0.45
Using CATALINA_HOME:   /home/stratebi/base/app/apache-tomcat-6.0.45
Using CATALINA_TMPDIR: /home/stratebi/base/app/apache-tomcat-6.0.45/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /home/stratebi/base/app/apache-tomcat-6.0.45/bin/bootstrap.jar
[stratebi@pentaho bin]$
```
Terminal

2. Arranque el BISERVER
   (a) Vaya al directorio **app > biserver-ce-5.4.0.1**, inicie el servidor utilizando el script **start-pentaho-ssl-trust.sh**. Luego presione la tecla **INTRO** para continuar con el arranque.

```
[stratebi@pentaho bin]$ cd -
[stratebi@pentaho base]$ cd ./app/biserver-ce-5.4.0.1
[stratebi@pentaho biserver-ce-5.4.0.1]$ ./start-pentaho-ssl-trust.sh
WARNING: Using java from path
DEBUG: _PENTAHO_JAVA_HOME=
DEBUG: _PENTAHO_JAVA=java
-------------------------------------------------------------------------------
The Pentaho BI Platform now contains a version checker that will notify you
```
Terminal

```
when newer versions of the software are available. The version checker is enabled by default.
For information on what the version checker does, why it is beneficial, and how it works see:
http://wiki.pentaho.com/display/ServerDoc2x/Version+Checker
Press Enter to continue, or type cancel or Ctrl-C to prevent the server from starting.
You will only be prompted once with this question.
-------------------------------------------------------------------------------------------
[OK]:

Using CATALINA_BASE:   /home/stratebi/base/app/biserver/tomcat
Using CATALINA_HOME:   /home/stratebi/base/app/biserver/tomcat
Using CATALINA_TMPDIR: /home/stratebi/base/app/biserver/tomcat/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /home/stratebi/base/app/biserver/tomcat/bin/bootstrap.jar
[stratebi@pentaho biserver-ce-5.4.0.1]$
```

3. Navegue a la dirección `http://localhost:8080/` utilizando navegador web de su preferencia. Esto hará una petición de conexión al BISERVER, el cual luego redirige la petición hacia el Servidor CAS para que realice la autenticación en su nombre: `https://localhost:8444/cas-server-webapp-3.5.3/login?service=http%3A%2F%2Flocalhost%3A8080%2Fpentaho%2Fj_spring_cas_security_check`
4. Dado a que el certificado utilizado no está firmado por ninguna autoridad reconocible por el navegador web, debemos agregar una excepción de seguridad para poder continuar hacia la página de ingreso del Servidor CAS.
5. Ingrese **admin** como usuario y contraseña. Esto es posible porque el Servidor CAS fue instalado en su configuración predeterminada así que este, otorgará autorización a toda credencial donde el **usuario sea igual a su contraseña**. El Servidor CAS validará las credenciales, otorga una ficha de autorización y redirige al navegador web devuelta al BISERVER.
6. El BISERVER recibe la ficha de autorización, valida la ficha contra el Servidor CAS y redirige el navegador a su página principal, `http://localhost:8080/pentaho/Home`, comprobando así, la correcta configuración del BISERVER con el Servidor CAS.

6 Anexos

6.1 Ficheros modificados durante la instalación

6.1.1 server.xml. Servidor CAS

Texto

```
<?xml version='1.0' encoding='utf-8'?>
<!--
  Licensed to the Apache Software Foundation (ASF) under one or more
  contributor license agreements.  See the NOTICE file distributed with
  this work for additional information regarding copyright ownership.
  The ASF licenses this file to You under the Apache License, Version 2.0
  (the "License"); you may not use this file except in compliance with
  the License.  You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.
-->
<!-- Note:  A "Server" is not itself a "Container", so you may not
     define subcomponents such as "Valves" at this level.
     Documentation at /docs/config/server.html
 -->
<Server port="8006" shutdown="SHUTDOWN">

  <!--APR library loader. Documentation at /docs/apr.html -->
  <Listener className="org.apache.catalina.core.AprLifecycleListener" SSLEngine="on" />
  <!--Initialize Jasper prior to webapps are loaded. Documentation at /docs/jasper-howto.html -->
  <Listener className="org.apache.catalina.core.JasperListener" />
  <!-- Prevent memory leaks due to use of particular java/javax APIs-->
  <Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener" />
  <!-- JMX Support for the Tomcat server. Documentation at /docs/non-existent.html -->
  <Listener className="org.apache.catalina.mbeans.ServerLifecycleListener" />
  <Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener" />

  <!-- Global JNDI resources
       Documentation at /docs/jndi-resources-howto.html
  -->
  <GlobalNamingResources>
    <!-- Editable user database that can also be used by
         UserDatabaseRealm to authenticate users
    -->
    <Resource name="UserDatabase" auth="Container"
              type="org.apache.catalina.UserDatabase"
              description="User database that can be updated and saved"
              factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
              pathname="conf/tomcat-users.xml" />
  </GlobalNamingResources>

  <!-- A "Service" is a collection of one or more "Connectors" that share
       a single "Container" Note:  A "Service" is not itself a "Container",
       so you may not define subcomponents such as "Valves" at this level.
       Documentation at /docs/config/service.html
   -->
  <Service name="Catalina">

    <!--The connectors can use a shared executor, you can define one or more named thread pools-->
    <!--
    <Executor name="tomcatThreadPool" namePrefix="catalina-exec-"
        maxThreads="150" minSpareThreads="4"/>
    -->


    <!-- A "Connector" represents an endpoint by which requests are received
         and responses are returned. Documentation at :
         Java HTTP Connector: /docs/config/http.html (blocking & non-blocking)
         Java AJP  Connector: /docs/config/ajp.html
         APR (HTTP/AJP) Connector: /docs/apr.html
         Define a non-SSL HTTP/1.1 Connector on port 8080
    -->
    <Connector port="8088" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8444" />
    <!-- A "Connector" using the shared thread pool-->
    <!--
    <Connector executor="tomcatThreadPool"
               port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000"
```

```
               redirectPort="8443" />
    -->
    <!-- Define a SSL HTTP/1.1 Connector on port 8443
         This connector uses the JSSE configuration, when using APR, the
         connector should be using the OpenSSL style configuration
         described in the APR documentation -->

    <Connector port="8444" protocol="HTTP/1.1" SSLEnabled="true"
               keystoreFile="cas-server.jks" keystorePass="password" keyAlias="serverkey"
               maxThreads="150" scheme="https" secure="true"
               clientAuth="false" sslProtocol="TLS" />

    <!-- Define an AJP 1.3 Connector on port 8009 -->
    <Connector port="8010" protocol="AJP/1.3" redirectPort="8444" />


    <!-- An Engine represents the entry point (within Catalina) that processes
         every request.  The Engine implementation for Tomcat stand alone
         analyzes the HTTP headers included with the request, and passes them
         on to the appropriate Host (virtual host).
         Documentation at /docs/config/engine.html -->

    <!-- You should set jvmRoute to support load-balancing via AJP ie :
    <Engine name="Catalina" defaultHost="localhost" jvmRoute="jvm1">
    -->
    <Engine name="Catalina" defaultHost="localhost">

      <!--For clustering, please take a look at documentation at:
          /docs/cluster-howto.html  (simple how to)
          /docs/config/cluster.html (reference documentation) -->
      <!--
      <Cluster className="org.apache.catalina.ha.tcp.SimpleTcpCluster"/>
      -->

      <!-- The request dumper valve dumps useful debugging information about
           the request and response data received and sent by Tomcat.
           Documentation at: /docs/config/valve.html -->
      <!--
      <Valve className="org.apache.catalina.valves.RequestDumperValve"/>
      -->

      <!-- This Realm uses the UserDatabase configured in the global JNDI
           resources under the key "UserDatabase".  Any edits
           that are performed against this UserDatabase are immediately
           available for use by the Realm.  -->
      <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
             resourceName="UserDatabase"/>

      <!-- Define the default virtual host
           Note: XML Schema validation will not work with Xerces 2.2.
       -->
      <Host name="localhost"  appBase="webapps"
            unpackWARs="true" autoDeploy="true"
            xmlValidation="false" xmlNamespaceAware="false">

        <!-- SingleSignOn valve, share authentication between web applications
             Documentation at: /docs/config/valve.html -->
        <!--
        <Valve className="org.apache.catalina.authenticator.SingleSignOn" />
        -->

        <!-- Access log processes all example.
             Documentation at: /docs/config/valve.html -->
        <!--
        <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"
               prefix="localhost_access_log." suffix=".txt" pattern="common" resolveHosts="false"/>
        -->

      </Host>
    </Engine>
  </Service>
</Server>
```

6.1.2 pentaho-spring-beans.xml. BISERVER

```
<?xml version='1.0' encoding='utf-8'?>                                                                    Texto
<!--+
  | This should be the only file specified in web.xml's contextConfigLocation. It should only contain imports.
  +-->

<beans xmlns="http://www.springframework.org/schema/beans" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:pen="http://www.pentaho.com/schema/pentaho-system"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
```

```
http://www.pentaho.com/schema/pentaho-system http://www.pentaho.com/schema/pentaho-system.xsd" default-lazy-
init="true">

<bean class="org.pentaho.platform.engine.core.system.objfac.spring.ApplicationContextPentahoSystemRegisterer"
scope="singleton"/>

  <bean id="SystemConfig" class="org.pentaho.platform.config.SystemConfig">
    <constructor-arg>
      <pen:list class="org.pentaho.platform.api.engine.IConfiguration"/>
    </constructor-arg>
    <pen:publish as-type="INTERFACES"/>
  </bean>

  <bean class="org.pentaho.platform.config.SolutionPropertiesFileConfiguration">
    <constructor-arg value="security"/>
    <constructor-arg value="security.properties"/>
    <pen:publish as-type="INTERFACES"/>
  </bean>

  <bean class="org.pentaho.platform.config.PentahoPropertyPlaceholderConfigurer" >
    <constructor-arg>
      <pen:bean class="org.pentaho.platform.api.engine.ISystemConfig"/>
    </constructor-arg>
  </bean>

  <bean class="org.pentaho.platform.config.SolutionPropertiesFileConfiguration">
    <constructor-arg value="system"/>
    <constructor-arg value="system.properties"/>
    <pen:publish as-type="INTERFACES"/>
  </bean>

  <bean class="org.pentaho.platform.config.SolutionPropertiesFileConfiguration">
    <constructor-arg value="sqlmetadataqueryexec"/>
    <constructor-arg value="sqlmetadataqueryexec.properties"/>
    <pen:publish as-type="INTERFACES"/>
  </bean>

  <import resource="pentahoSystemConfig.xml" />
  <import resource="adminPlugins.xml" />
  <import resource="systemListeners.xml" />
  <import resource="repository.spring.xml" />
  <import resource="applicationContext-spring-security.xml" />
  <import resource="applicationContext-spring-security-superuser.xml" />
  <import resource="applicationContext-pentaho-security-superuser.xml" />

  <import resource="applicationContext-common-authorization.xml" />
  <import resource="applicationContext-spring-security-memory.xml" />

  <import resource="applicationContext-pentaho-security-memory.xml" />
  <import resource="applicationContext-spring-security-ldap.xml" />
  <import resource="applicationContext-pentaho-security-ldap.xml" />

  <import resource="applicationContext-pentaho-security-jackrabbit.xml" />
  <import resource="applicationContext-spring-security-jackrabbit.xml" />
  <import resource="applicationContext-pentaho-security-jdbc.xml" />
  <import resource="applicationContext-spring-security-jdbc.xml" />
  <import resource="applicationContext-spring-security-cas.xml" />


  <import resource="pentahoObjects.spring.xml" />
  <import resource="GettingStartedDB-spring.xml" /> <!-- Remove this line to unhook the Getting Started DB -->
  <import resource="importExport.xml" />
  <import resource="defaultUser.spring.xml"/>
  <import resource="sessionStartupActions.xml" />
  <import resource="olap4j.spring.xml"/>
</beans>
```

6.1.3 applicationContext-spring-security-cas.xml. BISERVER

Texto

```
<?xml version='1.0' encoding='utf-8'?>

<beans xmlns="http://www.springframework.org/schema/beans" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:pen="http://www.pentaho.com/schema/pentaho-system"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
       http://www.pentaho.com/schema/pentaho-system http://www.pentaho.com/schema/pentaho-system.xsd" default-lazy-
init="true">

  <bean id="serviceProperties" class="org.springframework.security.ui.cas.ServiceProperties">
    <property name="service" value="http://localhost:8080/pentaho/j_spring_cas_security_check" />
    <property name="sendRenew" value="false" />
  </bean>

  <bean id="casProcessingFilter" class="org.springframework.security.ui.cas.CasProcessingFilter">
    <property name="authenticationManager" ref="authenticationManager" />
    <property name="authenticationFailureUrl" value="https://localhost:8444/cas-server-webapp-
3.5.3/authorizationFailure.jsp" />
    <property name="defaultTargetUrl" value="/" />
    <property name="filterProcessesUrl" value="/j_spring_cas_security_check" />
  </bean>

  <bean id="casProcessingFilterEntryPoint" class="org.springframework.security.ui.cas.CasProcessingFilterEntryPoint">
    <property name="loginUrl" value="https://localhost:8444/cas-server-webapp-3.5.3/login"/>
    <property name="serviceProperties" ref="serviceProperties" />
  </bean>

  <bean id="casAuthenticationProvider" class="org.springframework.security.providers.cas.CasAuthenticationProvider">
    <property name="userDetailsService">
      <pen:bean class="org.springframework.security.userdetails.UserDetailsService"/>
    </property>
    <property name="serviceProperties" ref="serviceProperties"/>
    <property name="ticketValidator">
      <bean class="org.jasig.cas.client.validation.Cas20ServiceTicketValidator">
        <constructor-arg index="0" value="https://localhost:8444/cas-server-webapp-3.5.3" />
      </bean>
    </property>
    <property name="key" value="an_id_for_this_auth_provider_only"/>
    <pen:publish as-type="org.springframework.security.providers.AuthenticationProvider">
      <pen:attributes>
        <pen:attr key="providerName" value="cas"/>
      </pen:attributes>
    </pen:publish>
  </bean>

   <bean id="filterChainProxy" class="org.springframework.security.util.FilterChainProxy">
    <property name="filterInvocationDefinitionSource">
      <value>
        <![CDATA[CONVERT_URL_TO_LOWERCASE_BEFORE_COMPARISON
        PATTERN_TYPE_APACHE_ANT

/webservices/**=securityContextHolderAwareRequestFilterForWS,httpSessionPentahoSessionContextIntegrationFilter,httpSess
ionContextIntegrationFilter,casProcessingFilter,basicProcessingFilter,anonymousProcessingFilter,exceptionTranslationFil
terForWS,filterInvocationInterceptorForWS

/api/**=securityContextHolderAwareRequestFilterForWS,httpSessionPentahoSessionContextIntegrationFilter,httpSessionConte
xtIntegrationFilter,casProcessingFilter,basicProcessingFilter,anonymousProcessingFilter,exceptionTranslationFilterForWS
,filterInvocationInterceptorForWS

/plugin/**=securityContextHolderAwareRequestFilterForWS,httpSessionPentahoSessionContextIntegrationFilter,httpSessionCo
ntextIntegrationFilter,casProcessingFilter,basicProcessingFilter,anonymousProcessingFilter,exceptionTranslationFilterFo
rWS,filterInvocationInterceptorForWS

/**=securityContextHolderAwareRequestFilter,httpSessionPentahoSessionContextIntegrationFilter,httpSessionContextIntegra
tionFilter,httpSessionReuseDetectionFilter,logoutFilter,casProcessingFilter,authenticationProcessingFilter,basicProcess
ingFilter,requestParameterProcessingFilter,anonymousProcessingFilter,exceptionTranslationFilter,filterInvocationInterce
ptor]]>
      </value>
    </property>
  </bean>

  <bean id="authenticationManager" class="org.springframework.security.providers.ProviderManager">
    <property name="providers">
      <list>
        <ref bean="casAuthenticationProvider" />
        <ref bean="anonymousAuthenticationProvider" />
      </list>
    </property>
  </bean>

  <bean id="exceptionTranslationFilter" class="org.springframework.security.ui.ExceptionTranslationFilter">
    <property name="authenticationEntryPoint">
      <ref local="casProcessingFilterEntryPoint" />
```

```
    </property>
    <property name="accessDeniedHandler">
      <bean class="org.springframework.security.ui.AccessDeniedHandlerImpl" />
    </property>
  </bean>

  <bean id="exceptionTranslationFilterForWS" class="org.springframework.security.ui.ExceptionTranslationFilter">
    <property name="authenticationEntryPoint">
      <ref local="casProcessingFilterEntryPoint" />
    </property>
    <property name="accessDeniedHandler">
      <bean class="org.springframework.security.ui.AccessDeniedHandlerImpl" />
    </property>
  </bean>

  <bean id="logoutFilter" class="org.springframework.security.ui.logout.LogoutFilter">
    <constructor-arg value="https://localhost:8444/cas-server-webapp-3.5.3/logout?
url=http://localhost:8080/pentaho/Home" />
    <!-- URL redirected to after logout -->
    <constructor-arg>
      <list>
        <bean class="org.pentaho.platform.web.http.security.PentahoLogoutHandler" />
        <bean
            class="org.springframework.security.ui.logout.SecurityContextLogoutHandler" />
      </list>
    </constructor-arg>
    <property name="filterProcessesUrl" value="/Logout" />
  </bean>
</beans>
```

6.1.4 start-pentaho-ssl-trust.sh. BISERVER

Texto

```
#!/bin/sh
### ====================================================================== ###
##                                                                          ##
##  Pentaho Start Script                                                    ##
##                                                                          ##
### ====================================================================== ###

DIR_REL=`dirname $0`
cd $DIR_REL
DIR=`pwd`
#cd -

. "$DIR/set-pentaho-env.sh"

setPentahoEnv "$DIR/jre"
errCode=0
if [ -f "$DIR/promptuser.sh" ]; then
  sh "$DIR/promptuser.sh"
  errCode="$?"
  rm "$DIR/promptuser.sh"
fi
if [ "$errCode" = 0 ]; then
  cd "$DIR/tomcat/bin"
  CATALINA_OPTS="-Xms1024m -Xmx2048m -XX:MaxPermSize=256m -Dsun.rmi.dgc.client.gcInterval=3600000
-Dsun.rmi.dgc.server.gcInterval=3600000 -Djavax.net.ssl.trustStorePassword=password
-Djavax.net.ssl.trustStore=/home/invitado/casguide/base/app/biserver-ce-5.4.0.1/cas-server-truststores.jks"
  export CATALINA_OPTS
  JAVA_HOME=$_PENTAHO_JAVA_HOME
  sh startup.sh
fi
```

6.2 Logs

6.2.1 catalina.out. Log de la primera corrida del BISERVER

Fichero Log

```
Mar 29, 2016 11:10:00 AM org.apache.catalina.core.AprLifecycleListener init
INFO: The APR based Apache Tomcat Native library which allows optimal performance in production
environments was not found on the java.library.path:
/usr/java/packages/lib/amd64:/usr/lib64:/lib64:/lib:/usr/lib
Mar 29, 2016 11:10:02 AM org.apache.coyote.http11.Http11Protocol init
INFO: Initializing Coyote HTTP/1.1 on http-8080
Mar 29, 2016 11:10:02 AM org.apache.catalina.startup.Catalina load
INFO: Initialization processed in 3738 ms
Mar 29, 2016 11:10:03 AM org.apache.catalina.core.StandardService start
INFO: Starting service Catalina
Mar 29, 2016 11:10:03 AM org.apache.catalina.core.StandardEngine start
INFO: Starting Servlet Engine: Apache Tomcat/6.0.43
Mar 29, 2016 11:10:03 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory ROOT
```

```
Mar 29, 2016 11:10:04 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory sw-style
Mar 29, 2016 11:10:05 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory pentaho-style
Mar 29, 2016 11:10:05 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory pentaho
[Server@1037449b]: [Thread[main,5,main]]: checkRunning(false) entered
[Server@1037449b]: [Thread[main,5,main]]: checkRunning(false) exited
[Server@1037449b]: Initiating startup sequence...
[Server@1037449b]: Server socket opened successfully in 7 ms.
[Server@1037449b]: Database [index=0, id=0, db=file:../../data/hsqldb/sampledata, alias=sampledata] opened
sucessfully in 3938 ms.
[Server@1037449b]: Database [index=1, id=1, db=file:../../data/hsqldb/hibernate, alias=hibernate] opened
sucessfully in 92 ms.
[Server@1037449b]: Database [index=2, id=2, db=file:../../data/hsqldb/quartz, alias=quartz] opened
sucessfully in 131 ms.
[Server@1037449b]: Startup sequence completed in 4173 ms.
[Server@1037449b]: 2016-03-29 11:10:18.075 HSQLDB server 1.8.0 is online
[Server@1037449b]: To close normally, connect and execute SHUTDOWN SQL
[Server@1037449b]: From command line, use [Ctrl]+[C] to abort abruptly
Attempting to load ESAPI.properties via file I/O.
Attempting to load ESAPI.properties as resource file via file I/O.
Not found in 'org.owasp.esapi.resources' directory or file not readable: /home/stratebi/base/app/biserver-
ce-5.4.0.1/tomcat/bin/ESAPI.properties
Not found in SystemResource Directory/resourceDirectory: .esapi/ESAPI.properties
Not found in 'user.home' (/home/stratebi) directory: /home/stratebi/esapi/ESAPI.properties
Loading ESAPI.properties via file I/O failed. Exception was: java.io.FileNotFoundException
Attempting to load ESAPI.properties via the classpath.
SUCCESSFULLY LOADED ESAPI.properties via the CLASSPATH from '/ (root)' using current thread context class
loader!
SecurityConfiguration for Validator.ConfigurationFile not found in ESAPI.properties. Using default:
validation.properties
Attempting to load validation.properties via file I/O.
Attempting to load validation.properties as resource file via file I/O.
Not found in 'org.owasp.esapi.resources' directory or file not readable: /home/stratebi/base/app/biserver-
ce-5.4.0.1/tomcat/bin/validation.properties
Not found in SystemResource Directory/resourceDirectory: .esapi/validation.properties
Not found in 'user.home' (/home/stratebi) directory: /home/stratebi/esapi/validation.properties
Loading validation.properties via file I/O failed.
Attempting to load validation.properties via the classpath.
validation.properties could not be loaded by any means. fail. Exception was:
java.lang.IllegalArgumentException: Failed to load ESAPI.properties as a classloader resource.
Pentaho BI Platform server is ready. (Pentaho Open Source BA Server 5.4.0.1-130) Fully Qualified Server
Url = http://localhost:8080/pentaho/, Solution Path = /home/stratebi/base/app/biserver-ce-5.4.0.1/pentaho-
solutions
Mar 29, 2016 11:17:08 AM org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-8080
Mar 29, 2016 11:17:09 AM org.apache.jk.common.ChannelSocket init
INFO: JK: ajp13 listening on /0.0.0.0:8009
Mar 29, 2016 11:17:09 AM org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=0/193  config=null
Mar 29, 2016 11:17:09 AM org.apache.catalina.startup.Catalina start
INFO: Server startup in 426472 ms
```

6.2.2 catalina.out. Log de la primera corrida del Servidor CAS

Fichero Log

```
Apr 06, 2016 10:35:04 AM org.apache.catalina.core.AprLifecycleListener init
INFO: The APR based Apache Tomcat Native library which allows optimal performance in production
environments was not found on the java.library.path:
/usr/java/packages/lib/amd64:/usr/lib64:/lib64:/lib:/usr/lib
Apr 06, 2016 10:35:05 AM org.apache.coyote.http11.Http11Protocol init
INFO: Initializing Coyote HTTP/1.1 on http-8088
Apr 06, 2016 10:35:07 AM org.apache.coyote.http11.Http11Protocol init
INFO: Initializing Coyote HTTP/1.1 on http-8444
Apr 06, 2016 10:35:07 AM org.apache.catalina.startup.Catalina load
INFO: Initialization processed in 3745 ms
Apr 06, 2016 10:35:07 AM org.apache.catalina.core.StandardService start
INFO: Starting service Catalina
Apr 06, 2016 10:35:07 AM org.apache.catalina.core.StandardEngine start
INFO: Starting Servlet Engine: Apache Tomcat/6.0.45
Apr 06, 2016 10:35:07 AM org.apache.catalina.startup.HostConfig deployDescriptor
INFO: Deploying configuration descriptor manager.xml
Apr 06, 2016 10:35:08 AM org.apache.catalina.startup.HostConfig deployDescriptor
INFO: Deploying configuration descriptor host-manager.xml
Apr 06, 2016 10:35:09 AM org.apache.catalina.startup.HostConfig deployWAR
INFO: Deploying web application archive cas-server-webapp-3.5.3.war
```

```
2016-04-06 10:35:25,391 INFO [org.jasig.cas.services.DefaultServicesManagerImpl] - <Loaded 1 services.>
2016-04-06 10:35:26,157 WARN
[org.jasig.cas.authentication.handler.support.SimpleTestUsernamePasswordAuthenticationHandler] -
<org.jasig.cas.authentication.handler.support.SimpleTestUsernamePasswordAuthenticationHandler is only to
be used in a testing environment.  NEVER enable this in a production environment.>
2016-04-06 10:35:30,606 INFO [org.jasig.cas.util.AutowiringSchedulerFactoryBean] - <Starting Quartz
Scheduler now>
Apr 06, 2016 10:35:35 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory ROOT
Apr 06, 2016 10:35:35 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory docs
Apr 06, 2016 10:35:36 AM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory examples
Apr 06, 2016 10:35:37 AM org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-8088
Apr 06, 2016 10:35:37 AM org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-8444
Apr 06, 2016 10:35:38 AM org.apache.jk.common.ChannelSocket init
INFO: JK: ajp13 listening on /0.0.0.0:8010
Apr 06, 2016 10:35:38 AM org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=0/147  config=null
Apr 06, 2016 10:35:38 AM org.apache.catalina.startup.Catalina start
INFO: Server startup in 31050 ms
2016-04-06 10:35:44,103 INFO [org.jasig.cas.ticket.registry.support.DefaultTicketRegistryCleaner] -
<Beginning ticket cleanup.>
2016-04-06 10:35:44,118 INFO [org.jasig.cas.ticket.registry.support.DefaultTicketRegistryCleaner] - <0
tickets found to be removed.>
2016-04-06 10:35:44,119 INFO [org.jasig.cas.ticket.registry.support.DefaultTicketRegistryCleaner] -
<Finished ticket cleanup.>
2016-04-06 10:37:25,755 INFO [org.jasig.cas.services.DefaultServicesManagerImpl] - <Reloading registered
services.>
2016-04-06 10:37:25,756 INFO [org.jasig.cas.services.DefaultServicesManagerImpl] - <Loaded 1 services.>
```

7 Sobre Stratebi

Stratebi Business Solutions es especialista en soluciones Business Intelligence, Analytics, Big Data, Liferay, soluciones Open Source, desarrollo Java y consultoría de Transformación Digital. Contactanos a través de:

web: ww.stratebi.com

email: info@stratebi.com

phone: (+34) 91.788.34.10

Estos son algunos ejemplos del portfolio de soluciones desarrolladas por Stratebi:


General
Efficiency
Economics
Performance
Savings Indicators
Consumption / Type of appliance
Energetic Status
Consumption / Hour type