

UD 3. Servidores de Aplicaciones (I)

Índice

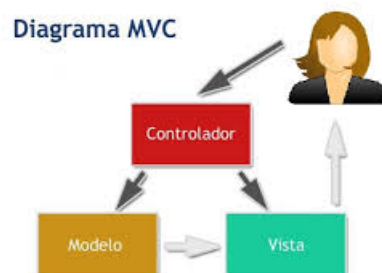
UD 3. Servidores de Aplicaciones (I).....	1
Arquitectura del servidor de aplicaciones.....	2
Servidores de aplicaciones existentes.....	2
Instalación de JDK (Java Development Tools).....	3
Instalación de Apache Tomcat 9.....	5
Instalación en Linux.....	5
Configurar la interfaz web de Tomcat.....	6
Archivos war.....	10
Despliegue de aplicaciones utilizando el 'manager-gui'.....	11
Configuración SSL sobre Tomcat.....	12

Arquitectura del servidor de aplicaciones.

Un servidor de aplicaciones permite dar acceso a las aplicaciones que se publican en el mismo, dando soporte para la seguridad, transacciones, administración de sesiones, registro de logs, etc.

La arquitectura empleada en aplicaciones a desplegar en un servidor de aplicaciones es el modelo MVC (Modelo-Vista-Controlador). Este modelo ha sido ampliamente adaptado como arquitectura para diseñar e implementar aplicaciones web, este patrón, como su nombre lo indica, utiliza tres componentes, modelo, vista, y controlador. Lo que hace este patrón es separar los datos y la lógica de negocio de la presentación y el módulo encargado de gestionar los eventos y las comunicaciones.

- **Modelo:** este componente representa la información con la cual el sistema opera, por lo tanto gestiona todos los datos, tanto consultas como actualizaciones, implementando también las reglas del negocio.
- **Vista:** en este componente solamente está las interfaces de usuario, ya sea formularios o archivos HTML. La vista se encarga de presentar la información del modelo, es decir, mostrar los datos que se solicita al modelo, en un formato adecuado, para luego mostrarlo en pantalla, a este componente se le conoce como salida.
- **Controlador:** como su nombre indica, se encarga de controlar (recibir las entradas), usualmente eventos que codifican los movimientos o pulsaciones de las teclas o botones del mouse, es decir, controla las acciones del usuario, por lo tanto, recibe las órdenes del usuario y se encarga de solicitar información al modelo y de comunicárselos a la vista.



Servidores de aplicaciones existentes.

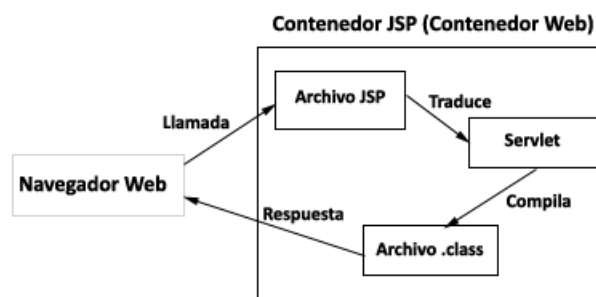
Los servidores de aplicaciones más utilizados son los siguientes:

- **Wildfly (Jboss Application Server):** implementado por Jboss y desarrollado en Java. Está disponible para todos los sistemas operativos del mercado y basado en proyectos de software open source.
- **GlassFish:** desarrollado por Sun Microsystems para la plataforma Java EE. Software de código abierto open source que permite tener disponibles las nuevas funcionalidades de Java.
- **JOnAs:** desarrollado por el consorcio ObjectWeb. Software libre.
- **Apache-Tomcat:** servidor de aplicaciones open source implementando Java Servlet, Java Servlet Pages (JSP), Java Expression Language y Java webSocket Technologies. Los anteriores módulos son desarrollados por Java Community Process.

Apache-Tomcat es un contenedor de Servlets que se puede usar para compilar y ejecutar aplicaciones realizadas en Java. Incluye el compilador Jasper, que compila JSPs convirtiéndolas en servlets. El motor de servlets de Tomcat a menudo se presenta en combinación con el servidor web Apache.

Un contenedor de **servlets** funciona del siguiente modo:

- El navegador del cliente solicita una página al servidor HTTP.
- El contenedor de servlets procesa la petición y le asigna el servlet apropiado.
- El servlet elegido es el encargado de generar el texto de la página web y entregarla al contenedor de servlets.
- El contenedor devuelve la página web al navegador del cliente.



Java Server Pages (**JSP**) es un software que permite generar páginas web dinámicas, así como otro tipo de documentos (como xml). Estos archivos .jsp se compilan y se transforman en un servlet.

Más adelante podremos ver los ejemplos de servlet y JSP que trae consigo el servidor Tomcat.

Instalación de JDK (Java Development Tools)

Tomcat necesita que tengamos instalada una versión de JDK. Nosotros utilizaremos Tomcat 9 , cuya versión mínima de JDK es la 8. Para poder empezar a hacer un uso completo de Apache Tomcat, necesitamos instalar una serie de paquetes necesarios. Podemos ver si ya tenemos alguna versión de JDK instalada con el comando siguiente que ejecutaremos utilizando el usuario root:

```
$ java -version
```

Aunque tenemos instalada una versión de Java, procederemos a descargar una versión compatible con Apache-Tomcat, para realizar posteriormente el proceso de configuración y selección de versiones instaladas. Por ejemplo, en nuestro caso, procedemos a la descarga desde la dirección <https://www.oracle.com/java/technologies/javase/javase-jdk8-downloads.html>, elegimos la opción de 64 bits y descargamos el fichero [jdk-8u311-linux-x64.tar.gz](#) o la versión existente.

A partir de ahora, utilizaremos un usuario con privilegios de administrador. Una vez descargado y colocado en el directorio `/opt`, creamos el directorio JDK dentro de `/opt` y lo descomprimos,

```
$ mkdir /opt/jdk
```

```
$ tar -zxf jdk-8u311-linux-x64.tar.gz -C /opt/jdk
```

A continuación hacemos que se configure la versión descargada de JDK en el sistema por defecto.

```
$ update-alternatives --install /usr/bin/java java /opt/jdk/jdk1.8.0_311/bin/java 100
```

```
$ update-alternatives --install /usr/bin/javac javac /opt/jdk/jdk1.8.0_311/bin/javac 100
```

Vamos a ejecutar los siguientes comandos para elegir la versión de Java a utilizar. Esto permitirá usar la versión descargada en Apache-Tomcat y cualquier aplicación que utilice Java.

```
$ update-alternatives --config java
```

```
$ update-alternatives --config javac
```

Si volvemos a ejecutar el comando, vemos que la versión utilizada por defecto ha cambiado.

Finalmente, **tras instalar el servidor Apache-Tomcat** (paso que se indica en el siguiente apartado), configuramos las variables de entorno JDK para que automáticamente se carguen en Apache-Tomcat cuando arranque el servicio.

Añadimos las variables de entorno utilizando el fichero `/etc/profile.d/tomcat9.sh`. Si no existe lo creamos.

```
sudo gedit /etc/profile.d/tomcat9.sh
```

Añadimos la líneas siguientes:

```
export JAVA_HOME=/opt/jdk/jdk1.8.0_311
export JRE_HOME=/opt/jdk/jdk1.8.0_311/jre
export PATH="$PATH:$JAVA_HOME"
export PATH="$PATH:$JRE_HOME"
```

El fichero `/etc/profile.d/tomcat9.sh` permitirá que cada vez que arranque la máquina, el componente JDK se cargue sin ser necesario configurarlo de manera manual. Podríamos haber llamado al fichero como `java.sh`. Aquí lo hemos llamado `tomcat9.sh`, ya que lo utilizaremos posteriormente y para reducir la cantidad de ficheros utilizados.

Cambiamos los permisos en el fichero, para añadir permisos de ejecución:

```
sudo chmod 755 /etc/profile.d/tomcat9.sh
```

La línea siguiente variará en función de la versión de Apache-Tomcat instalada.

```
$ /opt/tomcat9/apache-tomcat-9.0.54/bin/startup.sh
```

Podemos comprobar que se ha actualizado la ruta java utilizada.

También podemos instalar JDK de la siguiente forma:

```
$ sudo apt-get update
$ sudo apt-get install default-jdk
```

Instalación de Apache Tomcat 9

Instalación en Linux

El servidor de aplicaciones Tomcat no dispone de instalador por lo que la manera habitual de instalarlo es descargar la versión que queramos (descargaremos la versión 9), desde la web <https://tomcat.apache.org/download-90.cgi>, en *Binary Distributions* buscamos *Core* y descargamos el fichero comprimido *tar.gz*. Utilizaremos un usuario con privilegios de administrador. Colocamos dicho fichero en el directorio *opt*. Luego creamos un directorio llamado *tomcat9* dentro de *opt* y descomprimos el fichero con el siguiente comando (en lugar de *apache-tomcat-9.0.54.tar.gz* indicaremos el nombre de nuestro fichero). Está será la opción que elegiremos para instalar nuestro servidor Tomcat.

```
$ sudo tar xzf apache-tomcat-9.0.54.tar.gz -C /opt/tomcat9
```

Dentro de la carpeta `/opt/tomcat9/apache-tomcat-9.0.54` tendremos la instalación de Tomcat.

Como root, establecemos los permisos del directorio tomcat.

```
sudo chown -hR miusuario:miusuario /opt/tomcat9/apache-tomcat-9.0.54
sudo chmod 755 /opt/tomcat9/apache-tomcat-9.0.54/bin/*
```

Nota: En lugar de *miusuario* colocaremos el nombre de nuestro usuario.

Para iniciarlo, dispones de un script *startup.sh* que lo lanzará en segundo plano:

```
$ /opt/tomcat9/apache-tomcat-9.0.54/bin/startup.sh
```

Y también de un script *shutdown.sh* que lo detiene:

```
$ /opt/tomcat9/apache-tomcat-9.0.54/bin/shutdown.sh
```

Instalación en Debian utilizando apt

Podríamos instalar Tomcat desde los paquetes disponibles en los repositorios. t. Instalaríamos también algunos **paquetes extra**. Esta opción no es la que utilizaremos. Tenemos que tener en cuenta que, en este caso, todos los ficheros de configuración de *Apache Tomcat* se encuentran en */etc/tomcat9*.

```
$ sudo apt-get update
```

```
$ sudo apt-get install tomcat9
```

```
$ sudo apt-get install tomcat9-admin tomcat9-examples tomcat9-docs
```

También podríamos instalar las diferentes opciones utilizando:

```
$ sudo apt-get install tomcat9 tomcat9-*
```

Configurar la interfaz web de Tomcat

Aunque **la instalación no está todavía completa**, podemos echar un vistazo a la página principal de Apache Tomcat desde un navegador web utilizando la sintaxis **IP:PUERTO**. También podemos usar *127.0.0.1:8080* o *localhost:8080* si estamos probando en local. Como vemos, el puerto de escucha es el 8080, aunque se puede cambiar mediante el fichero *conf/server.xml*, ubicado dentro de la carpeta en la que hemos instalado nuestro servidor Tomcat.

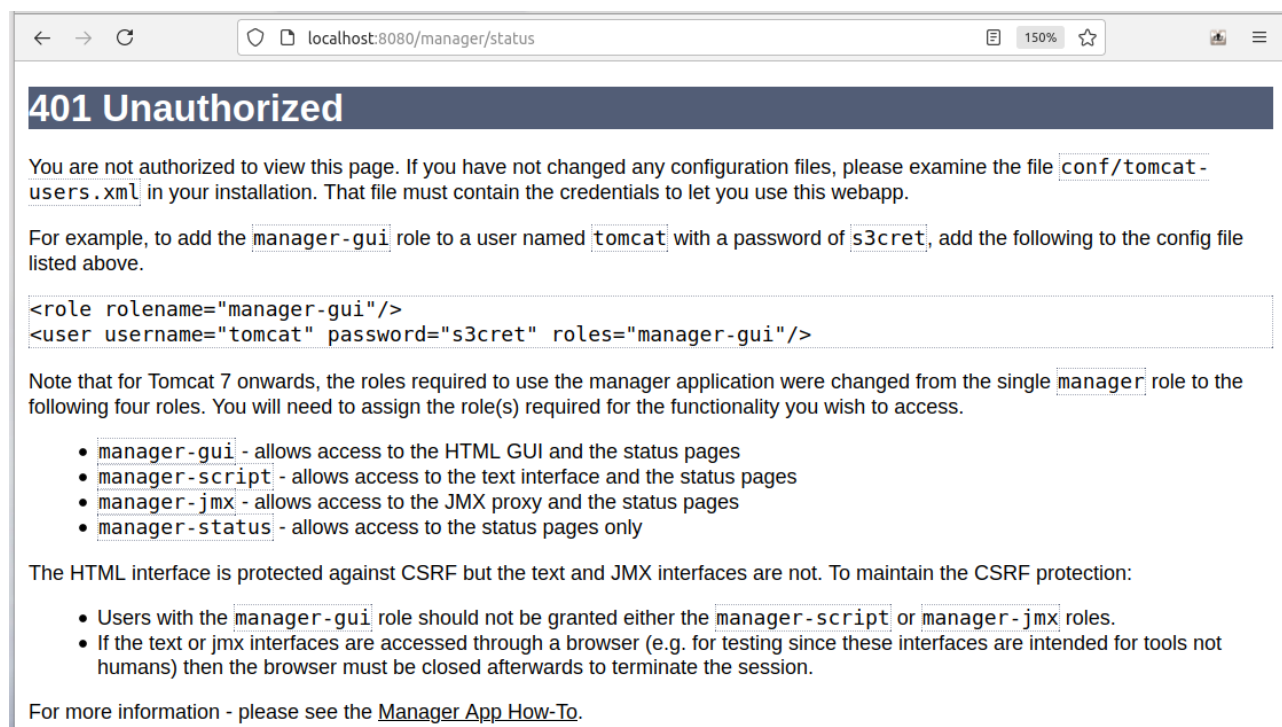
The screenshot shows the Apache Tomcat 9.0.54 web interface. At the top, there's a navigation bar with links: Home, Documentation, Configuration, Examples, Wiki, Mailing Lists, and Find Help. Below this, the title "Apache Tomcat/9.0.54" is displayed next to the Apache Software Foundation logo. A green banner reads: "If you're seeing this, you've successfully installed Tomcat. Congratulations!". Below the banner, there's a section for "Recommended Reading" with links to "Security Considerations How-To", "Manager Application How-To", and "Clustering/Session Replication How-To". To the right of these links are three buttons: "Server Status", "Manager App", and "Host Manager". Below this is a "Developer Quick Start" section with links to "Tomcat Setup", "First Web Application", "Realms & AAA", "JDBC DataSources", "Examples", "Servlet Specifications", and "Tomcat Versions". The main content area is divided into three columns: "Managing Tomcat" (with links to Release Notes, Changelog, Migration Guide, and Security Notices), "Documentation" (with links to Tomcat 9.0 Documentation, Tomcat 9.0 Configuration, Tomcat Wiki, and various developer resources), and "Getting Help" (with links to FAQ and Mailing Lists, and a list of available mailing lists). At the bottom, there's a navigation bar with links: Other Downloads, Other Documentation, Get Involved, Miscellaneous, and Apache Software.

Desde donde podemos acceder a diferentes secciones donde podemos monitorizar/configurar algunos aspectos del servidor.

Para configurar la interfaz Web lo primero que hay que hacer es crear los usuarios y los roles que queremos otorgarle. Por ejemplo, si en la página anterior clickamos en «*Server Status*», «*Manager App*» o «*Host Manager*» veremos que nos pide usuario y contraseña.

The screenshot shows the Tomcat Manager login page. The browser address bar shows "localhost:8080/manager". The page has a green background. In the center, there's a white box with the text "localhost:8080" and "Este sitio le pide que inicie sesión." Below this, there are two input fields: "Nombre de usuario" and "Contraseña". At the bottom of the white box, there are two buttons: "Cancelar" and "Iniciar sesión".

Si pulsamos en el botón *Cancelar*, Tomcat nos arroja un **Error 401 No Autorizado**. En esta pantalla podemos ver una ayuda de Tomcat indicando qué fichero necesitamos editar y qué líneas añadir para conseguir acceder a la configuración:



Esto es así, ya que, por seguridad, dicho acceso se encuentra restringido y tendremos que registrar usuarios en `conf/tomcat-users.xml` (ubicado dentro de la carpeta en la que hemos instalado nuestro servidor Tomcat) con nombre de usuario, contraseña y los roles con los que accederán.

Podemos crear un usuario para cada rol o bien un usuario que tenga varios de ellos (incluso todos). Cada rol definido por *Tomcat* da acceso a una de las partes del servidor:

- **manager-status:** Da acceso a la sección donde monitorizar el estado de Tomcat
- **manager-gui:** Da acceso al listado de aplicaciones y a poder desplegarlas desde la web
- **admin-gui:** Da acceso a la parte de administración de hosts virtuales

Editamos el archivo `tomcat-users.xml`, y dentro de la etiqueta `<tomcat-users>` creamos un usuario `tomcatuser` con contraseña `tomcatpsw` con todos los roles añadiendo:

```
<user username="tomcatuser" password="tomcatpsw" roles="manager-status, manager-gui, admin-gui"/>
```

Vemos que, desde el navegador, utilizando este usuario y contraseña, ahora tenemos acceso.

Por ejemplo, accediendo a «*Manager App*», vemos una fila con cada aplicación desplegada (podemos consultar la documentación o ejemplos), también nos da opción para desplegar aplicaciones, etc.

Gestor de Aplicaciones Web de Tomcat

Mensaje: OK

Gestor

[Listar Aplicaciones](#) [Ayuda HTML de Gestor](#) [Ayuda de Gestor](#) [Estado de Servidor](#)

Ruta	Versión	Nombre a Mostrar	Ejecutándose	Sesiones	Comandos
/	Ninguno especificado	Welcome to Tomcat	true	0	Arrancar <input type="button" value="Parar"/> <input type="button" value="Recargar"/> <input type="button" value="Replegar"/> <input type="button" value="Expirar sesiones"/> sin trabajar ≥ 30 minutos
/docs	Ninguno especificado	Tomcat Documentation	true	0	Arrancar <input type="button" value="Parar"/> <input type="button" value="Recargar"/> <input type="button" value="Replegar"/> <input type="button" value="Expirar sesiones"/> sin trabajar ≥ 30 minutos
/examples	Ninguno especificado	Servlet and JSP Examples	true	0	Arrancar <input type="button" value="Parar"/> <input type="button" value="Recargar"/> <input type="button" value="Replegar"/> <input type="button" value="Expirar sesiones"/> sin trabajar ≥ 30 minutos
/host-manager	Ninguno especificado	Tomcat Host Manager Application	true	0	Arrancar <input type="button" value="Parar"/> <input type="button" value="Recargar"/> <input type="button" value="Replegar"/> <input type="button" value="Expirar sesiones"/> sin trabajar ≥ 30 minutos
/manager	Ninguno especificado	Tomcat Manager Application	true	1	Arrancar <input type="button" value="Parar"/> <input type="button" value="Recargar"/> <input type="button" value="Replegar"/> <input type="button" value="Expirar sesiones"/> sin trabajar ≥ 30 minutos

Desde la página de [/examples](#) podremos ver una serie de ejemplos de Servlets, JSP y Websockets instalados con el paquete tomcat9-examples:

Apache Tomcat Examples - Mozilla Firefox

Apache Tomcat Examples

- [Servlets examples](#)
- [JSP Examples](#)
- [WebSocket Examples](#)

Tomcat ofrece una pantalla de configuración y gestión de las aplicaciones bastante completa y sencilla. Por ejemplo, si queremos que una aplicación deje de funcionar, es tan fácil como darle a “Parar” en su línea correspondiente. Si queremos volver a activarla, es tan sencillo como darle a “Arrancar”.

Archivos war

El archivo war es una de las partes más importantes de la aplicación web. El nombre WAR procede de Web Application Archive. Constituye una forma alternativa de distribuir aplicaciones Web empaquetando toda la aplicación (a partir de su directorio inicial) dentro de un fichero WAR (de forma parecida a como se hace con un TAR o un JAR), de modo que podemos distribuir dicho fichero en los servidores. El servidor debe ser compatible con servlets y páginas jsp, o cualquier lenguaje de programación usado en la aplicación web.

Podemos crear un fichero WAR de la misma forma que creamos un JAR, utilizando la herramienta JAR. Para ello, nos colocamos en el directorio de la aplicación y ejecutamos el comando siguiente

```
$ jar -cvf miapp.war *
```

También es posible generar el fichero WAR a través de cualquier IDE de programación.

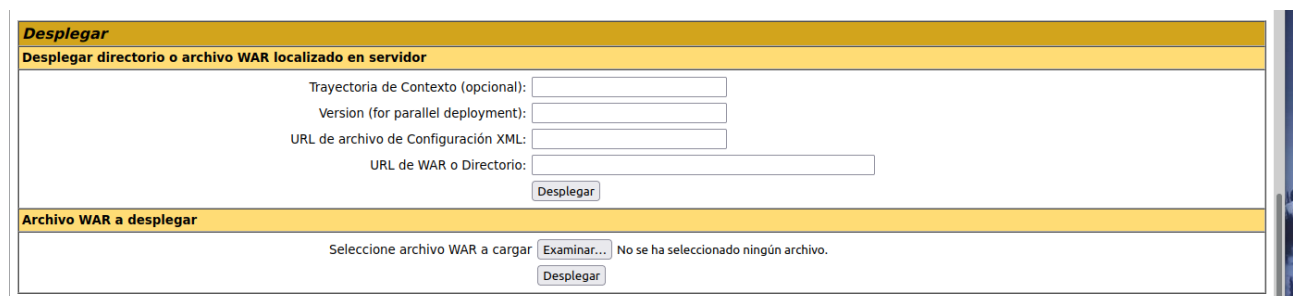
Estos ficheros WAR son un estándar de J2EE, por lo que podremos utilizarlos en los diferentes servidores de aplicaciones J2EE existentes.

El contenido del fichero puede ser el siguiente:

- Servlets, JSP o cualquier lenguaje de programación elegido para tal fin.
- Información web: XML, HTML, sonidos, vídeos, etc.
- Recursos web.

Despliegue de aplicaciones utilizando el 'manager-gui'

Se pueden desplegar aplicaciones web en Tomcat, de diversas maneras. La más sencilla es utilizando el propio servidor y accediendo al «manager-gui» de forma que podemos cargar el fichero .war de nuestra aplicación y lanzar la aplicación para su despliegue inmediato, como se muestra a continuación.



The screenshot displays the Tomcat Manager GUI with a yellow header bar labeled "Desplegar". Below the header, there are two main sections:

- Desplegar directorio o archivo WAR localizado en servidor**: This section contains four input fields: "Trayectoria de Contexto (opcional):", "Version (for parallel deployment):", "URL de archivo de Configuración XML:", and "URL de WAR o Directorio:". A "Desplegar" button is located at the bottom of this section.
- Archivo WAR a desplegar**: This section contains a label "Seleccione archivo WAR a cargar" followed by an "Examinar..." button and the text "No se ha seleccionado ningún archivo.". A "Desplegar" button is located at the bottom of this section.

Configuración SSL sobre Tomcat.

Lo ideal es utilizar el protocolo HTTPS, ya que permite la confidencialidad y la integridad de la información y además, se tiene asegurada la autenticación. Todo lo anterior utilizando la criptografía de clave pública.

Para usar transacciones SSL es necesario crear un almacén de claves. Para ello usaremos el programa de generación de claves de Java llamado keytool, incluido normalmente en el JDK.

Nos situamos en el directorio `opt/tomcat9/apache-tomcat-9.0.54/conf` y ejecutamos el comando siguiente:

```
keytool -genkey -alias tomcat -keyalg RSA -keystore tomcat.jks -validity 365 -keysize 2048
```

Mediante el comando anterior la opción alias permite identificar el fichero mediante el nombre `tomcat`. Decimos que el tipo de algoritmo es RSA, el almacen de claves es `tomcat.jks`, la validez 365 días y la clave de 2048 bits.

Introducimos la información que nos pide para crear el almacen de claves.

A continuación, editamos el fichero `server.xml`, ubicado en el directorio `conf` existente en la ubicación en la que hemos instalado el servidor Tomcat.

Lo editamos y buscamos la cabecera siguiente:

```
<!-- Define an SSL/TLS HTTP/1.1 Connector on port 8443
     This connector uses the NIO implementation. The default
     SSLImplementation will depend on the presence of the APR/native
     library and the useOpenSSL attribute of the
     AprLifecycleListener.
     Either JSSE or OpenSSL style configuration may be used regardless of
     the SSLImplementation selected. JSSE style configuration is used below.
-->
```

A continuación añadimos el código siguiente (donde en los atributos keystoreFile y keystorePass debes indicar los valores que se correspondan con el fichero .jks y password que has utilizado anteriormente), para establecer el conector que permite el uso de HTTPS.:

```
<Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"
           maxThreads="150" scheme="https" secure="true" SSLEnabled="true"
           KeystoreFile="conf/tomcat.jks" keystorePass="password" clientAuth="false"
           sslProtocol="TLSv1.2" SSLVerifyClient="none" >

</Connector>
```

Tras realizar los cambios anteriores, tendremos que detener y arrancar de nuevo el servidor Tomcat.

Finalmente, comprobamos en el navegador que podemos acceder mediante https. Para ello, introducimos la dirección <https://localhost:8443>

También podemos comprobar que accedemos a https accediendo directamente a la dirección web de las aplicaciones que tengamos instaladas.

Nos advierte que el sitio no es seguro, ya que el certificado utilizado no ha sido validado por ninguna entidad certificadora. La conexión sigue siendo segura.