Tema 1. Introducción a JEE

José Antonio Fajardo Naranjo
fajardonaranjoja@gmail.com

Resumen

Estos apuntes pertenecen al tema 1 de la asignatura de Desarrollo Web en entorno servidor impartida en el 2ndo curso de FPGS de DAW en el IES Martín Rivero durante el curso 2023/2024

Índice

1. Introducción

La **tecnología JEE** (Java Enterprise Edition) se presenta como una solución propuesta por Sun Microsystems para el desarrollo de aplicaciones distribuidas. Se **basa** en el lenguaje básico Java, también conocido como **JSE** (Java Standard Edition). JEE se puede considerar como una **normativa** que describe todos los elementos que constituyen e intervienen para el funcionamiento de una aplicación distribuida.

Define:

- Cómo se deben **desarrollar** los diferentes componentes de una aplicación (servlet, páginas JSP...).
- Cómo se deben **comunicar** con ellos o con otras aplicaciones (JDBC, JavaMail...).
- Cómo deben **organizarse** estos componentes para construir una aplicación (descriptor de despliegue).
- Las **restricciones** que tienen que respetar los servidores encargados de albergar estas aplicaciones.

El cumplimiento de esta normativa permite el desarrollo de servidores para las aplicaciones que respeten esta normativa, habiendo varios disponibles con rendimiento, capacidad y precio diferente.

La **ventaja** que ofrece JEE con **relación a tecnologías propietarias** (software privativo) reside en que cabe la posibilidad de evolucionar a otro servidor con más rendimiento sin necesidad de grandes modificaciones en la aplicación.

2. Servidores Web y servidores de aplicaciones

Un **servidor Web es un servidor de archivos**. Los clientes se dirigen a él mediante el **protocolo HTTP** para obtener un recurso. Cuando el servidor Web recibe la petición HTTP, extrae de la petición el recurso solicitado, lo busca en el disco y envía el recurso dentro de la respuesta HTTP para devolverlo al cliente. Esto se resume en que el **servidor Web no realiza ningún tipo de tratamiento** en el recurso antes de transmitirlo, no importando el tipo de recurso que le es solicitado. Este servidor se usa para recursos **estáticos**.

La función de un **servidor de aplicaciones** es la de **alojar el código** y **ejecutarlo** para, posteriormente, mediante el protocolo HTTP, **devolver** el resultado de la **ejecución** del código alojado de vuelta al cliente. Este servidor se utiliza para recursos **dinámicos**.

La **diferencia** entre el servidor Web y el de aplicaciones radica en esto, en que, mientras el **servidor Web** sólo se dedica a **enviar archivos** indistintamente según sea solicitado por el cliente, el **servidor de aplicaciones** se encarga de **ejecutarlo** en lugar del cliente devolviendo el resultado de la ejecución como texto plano.

La confusión viene dada porque en ocasiones el **servidor de aplicaciones también** toma las **funciones** de **servidor Web**. Cuando el servidor recibe una petición del **exterior**, la parte del servidor Web la **analiza**. Si la petición es de un recurso **estático**, el servidor Web cumple **su función** y se encarga de la petición, reenviando el recurso al cliente en una respuesta HTTP. Si, por el contrario, esa petición es de un recurso **dinámico**, el servidor Web la transfiere al **servidor de aplicaciones**, ejecutando este último el código correspondiente y generando la respuesta HTTP. En caso de ser necesario, el servidor de aplicaciones puede contactar con otro servidor o base de datos para la construcción de la respuesta, que se transmite al servidor Web, encargado de transmitirla al cliente de **vuelta**.

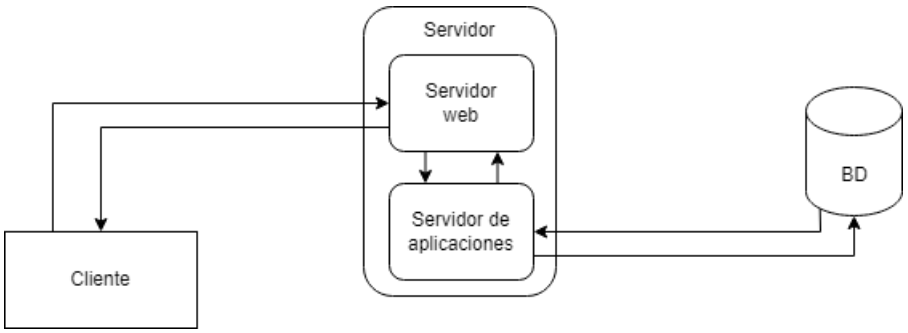

Figura 1: Explicación gráfica del entramado de solicitudes entre los servidores mostrado anteriormente

La parte que corresponde al servidor Web en un servidor de aplicaciones suele ser peor en cuanto a rendimiento que un servidor Web dedicado. Puede esta parte ser reemplazada por uno dedicado al adjuntarle un elemento llamado redirector, que transferirá las peticiones HTTP al servidor de aplicaciones que requieran recursos dinámicos, mientras que los estáticos los gestionará el propio servidor Web y el enlace entre redirector y servidor de aplicaciones lo proporcionará un protocolo de red propietario.

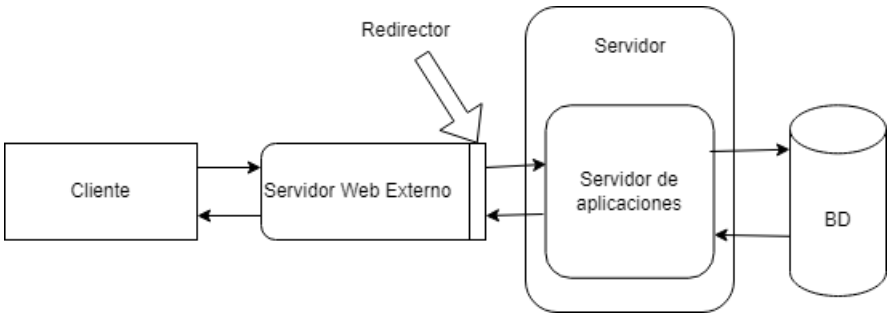

Figura 2: Ejemplo de solicitudes con un servidor Web dedicado

Se puede optar por esta solución en caso de que el servidor de aplicaciones esté altamente solicitado. Además, este redirector puede balancear las peticiones entre varios servidores de aplicaciones que sean idénticos

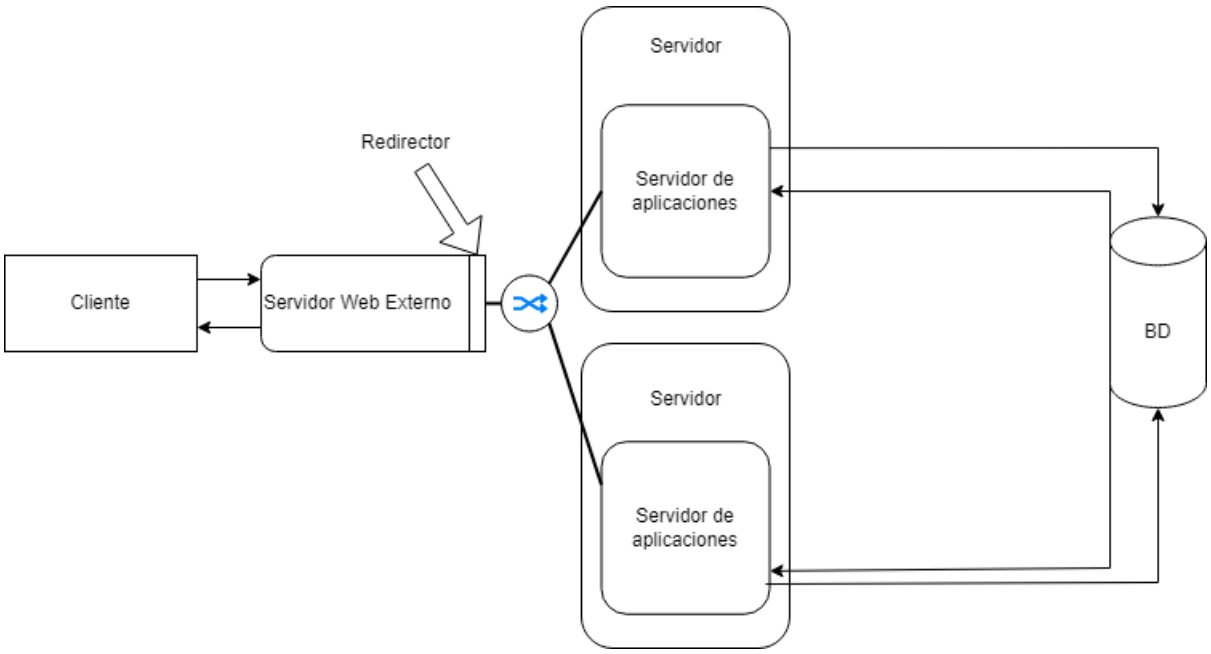

Figura 3: Ejemplo de solicitudes con un servidor Web dedicado y reparto de peticiones

3. Clientes ligeros y clientes pesados

Cuando se habla desde el punto de vista de las aplicaciones informáticas, se suele llamar cliente ligero a una solución que permite usar una aplicación desde un ordenador sin necesidad de instalación de ningún programa, función que realiza, por lo general, un navegador web. La mayoría de sistemas operativos llevan en su paquete este tipo de servicios. El servidor de aplicaciones se encarga del conjunto de tratamientos en este caso. La ventaja es la ausencia de intervención en los ordenadores cliente. La desventaja es la fuerte carga de solicitudes del servidor y la limitación del aspecto visual de la aplicación a las funcionalidades del lenguaje HTML, que se ayuda de otras tecnologías para mejorar la renderización gráfica de la aplicación.

El cliente pesado no tiene limitaciones en este ámbito. Además, permite aligerar la carga del servidor, que puede simplemente enviar al cliente los resultados en bruto de un tratamiento dejando al cliente la tarea de gestionar él mismo la presentación de estos resultados. La desventaja es el requisito de instalar la aplicación encargada del diálogo con el servidor y la presentación de los resultados en todos los ordenadores cliente. Cuando existe una nueva versión, también se ha de desplegar en todos los ordenadores cliente.

Pasamos por la mención de los applets, que se han quedado desactualizados.

4. Estructura de una aplicación JEE

La normativa JEE describe además, cómo se debe organizar una aplicación para que pueda ser soportada por cualquier servidor de aplicaciones compatible. Una aplicación web generalmente se compone de los siguientes elementos:

- **Recursos estáticos**: HTML, imágenes, sonidos, CSS...
- **Recursos dinámicos**: servlets, JSP, Java Bean.
- **Librerías** de clases utilizadas por los diferentes componentes dinámicos.
- **Descriptor de despliegue** que permita definir los parámetros de funcionamiento de la aplicación en el servidor, los enlaces entre las URL y los recursos dinámicos de la aplicación, las páginas por defecto y de error de la aplicación, la seguridad de la misma, etc.

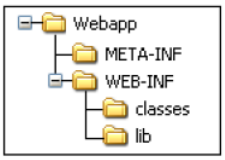

Figura 4: Árbol de organización de los elementos de una aplicación JEE

El directorio **Webapp** del ejemplo no hace que sea obligatorio que se llame así. Algunos servidores usan el nombre de la carpeta como nombre del servidor.

La **carpeta META-INF** contiene el MANIFEST.MF generado por la herramienta de archivado jar. Contiene la información descriptiva del archivo cuando la aplicación es desplegada con esta forma.

La **carpeta WEB-INF** contiene elementos únicamente accesibles por el servidor. Se encuentra el archivo web.xml, que es el descriptor de despliegue de la aplicación.

El subdirectorio **classes** del directorio WEB-INF contiene el código compilado de todas las clases necesarias para el funcionamiento de la aplicación. Si las clases están definidas en paquetes, esta carpeta ha de tener un árbol con estructura idéntica a la de los paquetes de la aplicación. Estos archivos nunca se transfieren a clientes, solo el servidor instancia las clases.

El subdirectorio **lib** de WEB-INF contiene las librerías para el funcionamiento de la aplicación. Estas se ubican generalmente en esta carpeta en forma de archivo Java (jar). Se puede encontrar, por ejemplo, la librería que contiene un driver de acceso a una BD o librerías de etiquetas JSP personalizadas.

5. Empaquetado de una aplicación

Para facilitar el despliegue de una aplicación se puede incluir todo el conjunto de archivos necesarios para el funcionamiento de la aplicación en un archivo empaquetado Java (.jar o .war). Este archivo con extensión .war se despliega automáticamente en el servidor.

La manipulación de un archivo empaquetado Java utiliza la sintaxis de manipulación de archivos muy similar a Unix. Estos se epaquetan internamente en formato ZIP.

5.1. Creación de un empaquetado

Es importante que antes de empaquetar se apague el servidor, ya que al hacerlo, automáticamente desplegará la aplicación desde ese empaquetado, pudiendo corromper la aplicación. La estructura para poder empaquetar una aplicación JEE es la siguiente:

```
jar cfv program.war -C program/ .
```

- El parámetro “c” sirve para indicar al comando jar que se desea crea¡r un empaquetado.
- El parámetro “f”, por su parte, indica que el comando tiene que generar un archivo.
- La tercera parte, el parámetro “v” muestra el nombre de los archivos a medida que se van añadiendo al empaquetado.
- Si queremos añadir más de un archivo ponemos los nombres separados por un espacio. El carácter comodín ‘*’ también está permitido en la lista. Si un nombre de directorio está presente en la lista, todo su contenido se añade al empaquetado.

5.2. Visualización del contenido

El contenido de un empaquetado puede visualizarse con el comando siguiente:

```
jar tf nombre.war
```

El comando muestra en consola el contenido del empaquetado. Se puede obtener información adicional, como la fecha de modificación y el tamaño del archivo, añadiendo la opción v al comando:

```
jar tvf nombre.war
```

Las rutas de acceso a los archivos, al estar en Windows, se muestran con el carácter / como separador, siendo relativas a la raíz del archivo. El contenido del empaquetado no es modificado por la ejecución del comando, ya que solo lo estamos visualizando.

5.3. Extracción

Los archivos pueden ser extraídos del empaquetado con el siguiente comando:

```
jar xvf nombre.war
```

Al extraer estos archivos se crean en disco en el directorio actual, sobreescribiendo los existentes que puedan ser reemplazados. No hay modificación de los archivos que se encuentran en el empaquetado. Si el archivo contiene una ruta, se crea de nuevo en el directorio actual. Podemos ser más específicos, extrayendo únicamente un archivo concreto del .war. A continuación un ejemplo con el archivo web.xml:

```
jar xvf nombre.war WEB-INF/web.xml
```

5.4. Actualización

El contenido de un empaquetado se puede actualizar mediante la adición de archivos después de su creación. El comando es el siguiente:

```
jar uf nombre.war imágenes/logo.gif
```

El último parámetro de este comando representa la lista de archivos que se actualizarán en el archivo. Si estos archivos no existían se añaden, en caso contrario, se reemplazan por la nueva versión. Si el empaquetado contiene directorios, la ruta de acceso completa tiene que especificarse en la lista de archivos que se añadirán.