Implantación de arquitecturas web

Aspectos generales de arquitecturas web.

La arquitectura de aplicaciones en entornos web difiere bastante de la de aplicaciones de escritorio, en la cual un programa se ejecuta directamente sobre la máquina en la que trabaja el usuario.

El modelo de arquitectura básico que existe en toda **aplicación web** es el modelo llamado cliente-servidor, en el cual entran en juego diversas máquinas o plataformas, cada una de las cuales desarrolla un rol diferenciado en la ejecución de la aplicación. Según las necesidades y la complejidad de la aplicación, este modelo básico de arquitectura puede complicarse más o menos para lograr una mejor distribución de tareas, mejor rendimiento, fiabilidad, aumento de la capacidad de proceso, etc.

Arquitecturas web. Modelos

Una aplicación distribuida está compuesta por una colección de ordenadores autónomos enlazados por una red de ordenadores y respaldados por un software que hace el conjunto actúe como un servicio integrado.

El modelo cliente-servidor

El modelo cliente-servidor es un modelo de arquitectura de aplicaciones en el cual se definen o se asignan principalmente dos roles a los ordenadores, que son, como el nombre del modelo indica, los roles de cliente y de servidor.

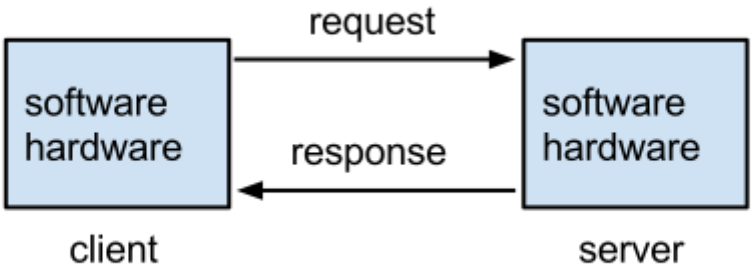

Imagen: Lubaochuan, CC BY-SA 4.0 Wikimedia Commons

En el modelo cliente-servidor, existen dos tipos de componentes:

- **Clientes**: realizan peticiones de servicio. Por lo general, los clientes inician la comunicación con el servidor.
- **Servidores**: proveen servicios. Normalmente, los servidores esperan recibir peticiones. Una vez que han recibido una petición, la resuelven y devuelven el resultado al cliente.

El modelo cliente-servidor básico de la figura anterior es válido para aplicaciones web pequeñas, simples y que no tengan una gran carga de trabajo, es decir, un número reducido de clientes conectados simultáneamente.

En entornos reales, es común que estas tres características no estén presentes, por lo que es necesario implementar una arquitectura más compleja pero también basada en el modelo cliente-servidor. Esta arquitectura puede presentar diferencias o extensiones al modelo básico para garantizar un buen rendimiento de las aplicaciones web, su fiabilidad y/o la capacidad de atender un gran número de peticiones de los clientes de forma simultánea en aplicaciones web de tamaño mediano o grande, y con un nivel de complejidad medio/alto. De esta necesidad surge el modelo siguiente.

Modelo Cliente-Servidor con Servidores Encadenados

Cuando en una aplicación el servidor debe realizar tareas muy complejas o costosas de procesar, se pueden distribuir subtareas en varios servidores. De esta manera, un servidor puede actuar como cliente de otro servidor para delegar ciertas responsabilidades.

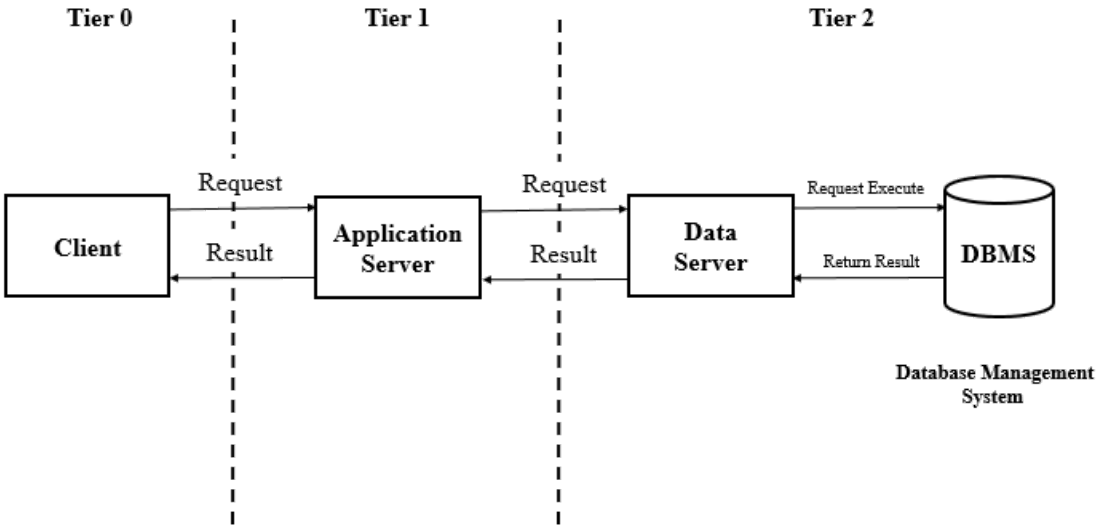

Imagen: Michel Bakni, CC BY-SA 4.0 Wikimedia Commons

Por ejemplo, cuando un cliente de una entidad bancaria accede a los servicios en línea de su banco a través de un navegador web (cliente), el cliente inicia una solicitud al servidor web del banco. Las credenciales de inicio de sesión del cliente se almacenan en una base de datos y el servidor web accede a la base de datos como cliente. Un servidor de aplicaciones interpreta los datos devueltos aplicando la lógica de negocios del banco y proporciona la salida al

servidor web. Finalmente, el servidor web devuelve el resultado al navegador web del cliente para su visualización.

Aplicaciones Basadas en la Web

Un caso particular de aplicaciones cliente-servidor son las aplicaciones que se ejecutan aprovechando la arquitectura web. Estas aplicaciones se basan en tener toda la capacidad de procesamiento en un servidor web (o conjunto de servidores) al que se accede desde un navegador web.

Cuando un usuario hace clic en un enlace en una página web de su navegador, este genera una solicitud al servidor que contiene la información. Una vez que el servidor recibe la solicitud, devuelve el contenido. La comunicación entre el cliente y el servidor se realiza a través del protocolo HTTP.

Modelo Peer-to-Peer (P2P)

Existe un tipo de arquitectura en la cual todas las computadoras actúan simultáneamente como clientes y servidores. Estas redes se conocen como redes peer-to-peer (igual a igual).

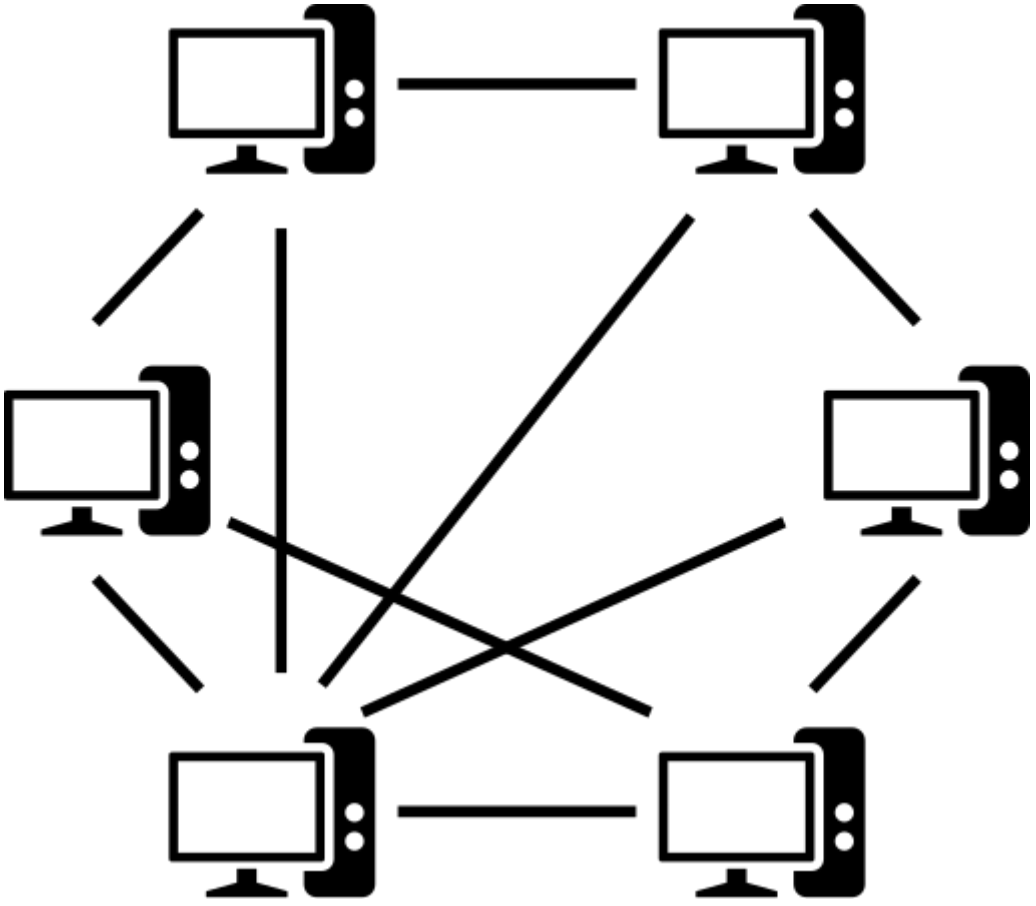

Imagen: The 360 Degree, CC BY-SA 4.0 Wikimedia Commons

Un sistema peer-to-peer se caracteriza por ser un sistema distribuido en el cual todos los nodos tienen las mismas capacidades y responsabilidades, es decir, todos son clientes y servidores al mismo tiempo, lo que implica que toda la comunicación es simétrica.

Servidores web y de aplicaciones. Instalación y configuración básica

Durante las fases de desarrollo, puesta en producción y mantenimiento de una aplicación web, nos encontramos con varios tipos de servidores que llevan a cabo tareas específicas en el funcionamiento global.

Servidores web

Un servidor web es un servidor que permite el acceso a recursos a través del protocolo HTTP (Protocolo de Transferencia de Hipertexto) de internet.

La definición original y estricta del concepto de servidor HTTP se refiere a aquellos servidores capaces de proporcionar acceso y permitir la gestión de un conjunto de recursos estáticos como respuesta a las peticiones recibidas de los clientes. Es decir, que permiten consultar, cargar y eliminar recursos del servidor. Estos recursos suelen ser documentos HTML o variantes de este formato y contenidos adjuntos o relacionados con estos documentos, como imágenes, videos, etc.

Estos recursos suelen estar guardados en forma de archivos en dispositivos de almacenamiento propios del servidor.

El concepto original de servidor web no contempla la posibilidad de generar de forma dinámica los contenidos a partir de la ejecución de código como respuesta a las peticiones. Sin embargo, en la actualidad, la mayoría de los servidores web admiten la instalación de módulos que permiten generar contenidos dinámicos a partir de la ejecución de programas escritos en diversos lenguajes de programación (PHP, JavaScript, Python, Perl, etc.), aunque esta característica es más propia de los servidores de aplicaciones.

Algunos ejemplos de servidores web son Apache o Nginx, para sistemas operativos Linux, y Microsoft Internet Information Server, para Windows.

Servidores de aplicaciones

Un servidor de aplicaciones, en general, es un servidor que ofrece a los clientes un servicio de ejecución de aplicaciones. Si nos centramos en las aplicaciones web, un servidor de aplicaciones es un software que controla la ejecución de programas. Los clientes, desde un navegador (usando el protocolo HTTP), acceden a una interfaz web desde donde ejecutarán

la aplicación. Normalmente, los servidores de aplicaciones se utilizan en aplicaciones web con un alto grado de complejidad.

Un servidor de aplicaciones web se puede entender como un servidor orientado a la ejecución de programas que puede recibir las peticiones de servicio y devolver los resultados utilizando los mismos protocolos (HTTP) y formatos de datos que los servidores web (HTML). Si el mismo servidor no tiene la capacidad de interactuar con estos protocolos, puede trabajar conjuntamente con el soporte de un servidor web que haga de intermediario entre el servidor de aplicaciones y el cliente. Los servidores de aplicaciones, además, suelen proporcionar un amplio conjunto de servicios complementarios orientados a la persistencia de datos, seguridad, control de transacciones y concurrencia, entre otros.

Algunos ejemplos de servidores de aplicaciones son GlassFish (servidor Java EE, Oracle), Tomcat o Microsoft Internet Information Server (servidor .NET).

Servidores de bases de datos

Un servidor de bases de datos se utiliza para almacenar, recuperar y administrar los datos de una base de datos. El servidor gestiona las actualizaciones de datos, permite el acceso simultáneo de muchos servidores o usuarios web y garantiza la seguridad y la integridad de los datos.

Entre sus funciones básicas, el software de servidores de bases de datos ofrece herramientas para facilitar y acelerar la administración de bases de datos. Algunas funciones son la exportación de datos, la configuración del acceso de usuarios y el soporte de datos.

Algunos ejemplos de servidores de bases de datos son Oracle Database, MySQL, Microsoft SQL Server, PostgreSQL, MongoDB o Firebase.

Servidores de archivos

Un servidor de archivos es un servidor que permite gestionar a través de red la carga, descarga, actualización y eliminación de archivos almacenados en sus dispositivos desde computadoras cliente.

En el ámbito de las aplicaciones web, los servidores de archivos se utilizan principalmente para implementar las aplicaciones en el servidor donde se ejecutarán. La implementación de una aplicación web en los servidores de producción generalmente implica la carga de grandes cantidades de archivos en estos servidores. Dado que el desarrollo y mantenimiento de estas aplicaciones se realiza en las máquinas de los programadores, se necesita un sistema de transferencia de archivos cada vez que se quiere actualizar la versión de producción de una aplicación.

Uno de los protocolos más utilizados para la transferencia de archivos en la implementación de aplicaciones web es el protocolo FTP (Protocolo de Transferencia de Archivos), con sus variantes FTPS y SFTP para adaptarse a las necesidades actuales de seguridad.

Algunos ejemplos de servidores de transferencia de archivos son ProFTPD o vsftpd, para sistemas operativos Linux, y Microsoft Internet Information Server, para Windows.

Servidores de directorio

Un servidor de directorio es un servidor que permite gestionar información administrativa sobre el entorno de una aplicación web, como pueden ser, por ejemplo, los usuarios autorizados con sus roles o permisos, etc.

La utilidad principal de los servidores de directorio es facilitar la gestión de información relacionada con la explotación de aplicaciones web. La ventaja de gestionar esta información mediante este tipo de servidores es la centralización de datos y la facilidad de acceso mediante protocolos estándar como LDAP.

Algunos ejemplos de servidores de directorio son OpenLDAP, para Linux, y Active Directory, para Windows.

Estructura y Recursos de una Aplicación Web

Las aplicaciones web, además de presentar una arquitectura cliente-servidor (hecho que no es necesario en el caso de las aplicaciones de escritorio), suelen estar estructuradas con una gran cantidad de archivos y recursos de diferentes tipos.

Por esta razón, es necesario establecer directrices para organizar la ubicación de estos componentes y su interrelación durante la fase de desarrollo, así como también al momento de poner la aplicación en producción. De lo contrario, el desarrollo y mantenimiento de una aplicación de tamaño mediano o grande se convertirá en una tarea casi imposible de gestionar.

Dejando de lado la organización o estructura impuesta por la elección de ciertas herramientas de desarrollo o un servidor web o de aplicaciones específico, estas aplicaciones se pueden estructurar según varios modelos de organización de sus componentes y recursos. Algunos de los modelos de estructuración de aplicaciones web que podemos encontrar más comúnmente son los que se describen a continuación.

Arquitectura Multinivel

La arquitectura multinivel (multitier architecture) es un tipo específico de la arquitectura cliente-servidor en la cual los componentes y recursos de una aplicación se separan según su

función. Una de las divisiones más utilizadas es la que separa el nivel de presentación, el nivel de lógica de aplicación y el nivel de gestión de datos.

En este caso, la estructura concreta sería de tres niveles (3-tier architecture). El modelo se define como N-tier architecture (multinivel), ya que propone una división flexible de las aplicaciones en los niveles que sean necesarios para hacer más eficiente su desarrollo, mantenimiento y explotación.

En este modelo, la división por niveles se realiza de forma lineal: el nivel 1 interactúa de forma directa y única con el nivel 2, el nivel 2 interactúa con el 3, y así sucesivamente.

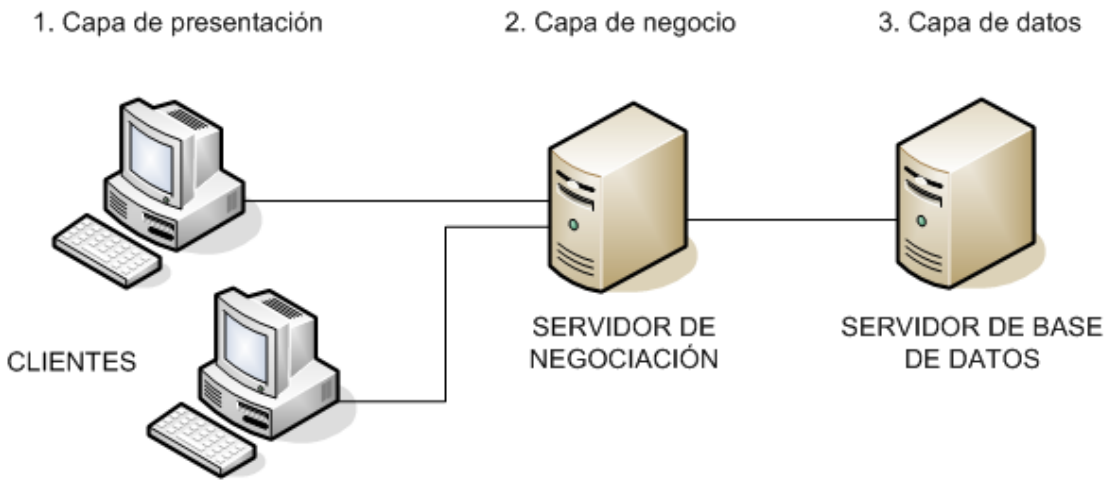

Imagen: No machine-readable author provided. Master Will assumed (based on copyright claims)., Dominio público Wikimedia Commons

Warning

Es importante diferenciar entre el concepto de multinivel (multitier o N-tier) y multicapa (multilayer o N-layer). En el caso del modelo multinivel, se considera que cada nivel, además de implementar una función concreta, es ejecutado por un hardware diferente al del resto de los niveles. En el modelo multicapa, cada capa desarrolla una función concreta que puede ser ejecutada por una misma computadora que se encarga también de la ejecución de otras capas.

Arquitectura Modelo-Vista-Controlador

La arquitectura Modelo-Vista-Controlador (Model-View-Controller o MVC) es una arquitectura que separa la representación de la información y la lógica de una aplicación de la interacción del usuario.

Los tres elementos que definen esta arquitectura son:

- **Modelo:** Contiene los datos de la aplicación, las reglas de negocio o la lógica de la aplicación y sus funciones.

- **Vista:** Es la representación visible de la aplicación, la salida de los datos hacia el usuario, es decir, la interfaz.
- **Controlador:** Controla la interacción del usuario (entrada de datos) y convierte esta interacción en órdenes o comandos para el modelo o la vista.

La interrelación entre los elementos de esta arquitectura no sigue un modelo lineal como el modelo multinivel, sino que se trata de un modelo circular.

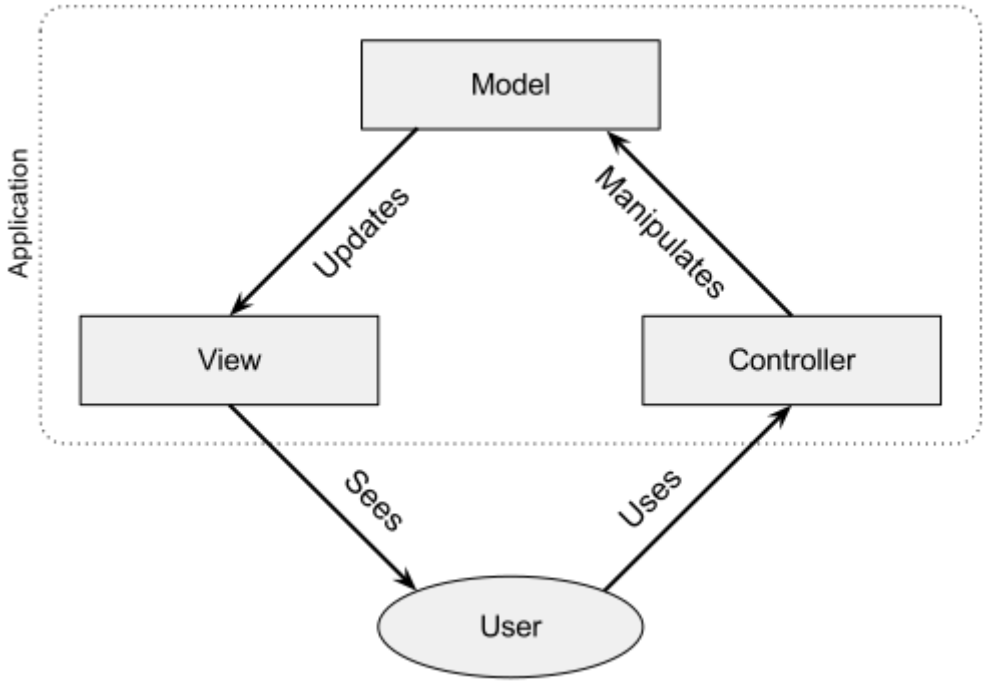

Imagen: Behnam Esfahbod, CC BY-SA 3.0 Wikimedia Commons

Paralelamente a la estructura de la aplicación, es necesario considerar que cada nivel, capa o módulo puede estar compuesto por un gran número de componentes y recursos de diversos tipos: archivos HTML, CSS, imágenes, etc.

Por ello, es conveniente establecer un sistema de organización coherente y eficiente para estructurar todos estos componentes que se generan durante el desarrollo de una aplicación web. La mayoría de las plataformas de desarrollo avanzadas imponen mecanismos para organizar y describir de manera sistemática la ubicación, características y configuración de los componentes y recursos de las aplicaciones.

Entre estos mecanismos, se destacan dos:

Estructura de Directorios

Las plataformas avanzadas de desarrollo de aplicaciones web suelen definir una estructura de directorios mínima que toda aplicación debe tener, a partir de la cual se despliegan los diferentes tipos de componentes. Los desarrolladores deben seguir las directrices de cada plataforma.

Descriptor de Despliegue

Existe un archivo de configuración en el cual se puede especificar el nombre, ubicación y parámetros de configuración de los diferentes componentes que conforman una aplicación. Esto permite tener dicha información centralizada, accesible y actualizable sin necesidad de modificar el código fuente de la aplicación. Este descriptor describe cómo se debe desplegar la aplicación en el servidor.

Plataformas Web libres y propietarias

Una plataforma web es el entorno de desarrollo de software empleado para diseñar y ejecutar un sitio web. En términos generales, una plataforma web consta de cuatro componentes básicos:

1. **Sistema operativo:** Bajo el cual opera el equipo donde se hospedan las páginas web y que representa la base misma del funcionamiento del computador. En ocasiones limita la elección de otros componentes.
2. **Servidor web:** El software que maneja las peticiones desde equipos remotos a través de la Internet. En el caso de páginas estáticas, el servidor web simplemente provee el archivo solicitado, el cual se muestra en el navegador. En el caso de sitios dinámicos, el servidor web se encarga de pasar las solicitudes a otros programas que puedan gestionarlas adecuadamente.
3. **Gestor de bases de datos:** Se encarga de almacenar sistemáticamente un conjunto de registros de datos relacionados para ser usados posteriormente.
4. **Lenguaje de programación interpretado:** Controla las aplicaciones de software que corren en el sitio web.

Diferentes combinaciones de los cuatro componentes señalados, basadas en las distintas opciones de software disponibles en el mercado, dan lugar a numerosas plataformas web, aunque, sin duda, hay dos que sobresalen del resto por su popularidad y difusión: **LAMP** y **WISA**.

Plataforma LAMP

La plataforma LAMP trabaja enteramente con componentes de software libre y no está sujeta a restricciones propietarias. El nombre LAMP surge de las iniciales de los componentes de software que la integran:

- **Linux:** sistema operativo.
- **Apache:** servidor web.
- **MySQL:** gestor de bases de datos.
- **PHP:** lenguaje interpretado PHP, aunque a veces se sustituye por Perl o Python.

Plataforma WISA

La plataforma WISA está basada en tecnologías desarrolladas por la compañía Microsoft; se trata, por lo tanto, de software propietario. La componen los siguientes elementos:

- **Windows:** sistema operativo.
- **Internet Information Services (IIS):** servidor web.
- **SQL Server:** gestor de bases de datos.
- **ASP o ASP.NET:** como lenguaje para scripting del lado del servidor.

Otras plataformas

Existen otras plataformas, como por ejemplo la configuración **WAMP** (Windows-Apache-MySQL-PHP), que es bastante común pero sólo como plataforma de desarrollo local. De forma similar, un servidor Windows puede correr con IIS y con MySQL y PHP. A esta configuración se la conoce como plataforma **WIMP**.

Entre los servidores web más utilizados hoy en día, aparte de Apache e IIS, tenemos también a **Nginx**, un servidor web software libre que está demostrando un alto rendimiento. Nginx es capaz de atender una gran cantidad de peticiones simultáneas y tiene un mejor rendimiento que sus competidores al servir contenido estático. Sin embargo, su configuración es menos flexible que otros.

Nginx también puede integrarse con PHP, dando lugar a plataformas tipo **LNMP** (Linux + Nginx + MySQL o MariaDB + PHP). Esta opción es también posible en Windows, dando lugar a plataformas **WNMP** (Windows + Nginx + MySQL o MariaDB + PHP).

Existen muchas otras plataformas que trabajan con distintos sistemas operativos (Unix, MacOS, Solaris), servidores web (incluyendo algunos que se han cobrado relativa popularidad como Lighttpd y LiteSpeed), bases de datos (MariaDB, PostgreSQL, MongoDB) y otros lenguajes de programación.

XAMPP: XAMPP es una forma fácil de instalar y usar el servidor web Apache con un sistema gestor de bases de datos (MariaDB), PHP y Perl. Basta con descargarlo, extraerlo y comenzar. En este momento hay cuatro versiones de XAMPP para Linux, Windows, Mac OS X y Solaris.

Escalabilidad

Las aplicaciones web se ejecutan en un entorno donde el número de clientes que solicitan el servicio puede variar en gran medida en función del momento. Es por ello que hay una característica de esencial importancia como es la **escalabilidad**.

En el entorno en que se ubican las aplicaciones web, uno de los principales factores que puede afectar al rendimiento de las mismas es el número de usuarios, ya que este puede verse incrementado de forma vertiginosa en un periodo de tiempo relativamente corto. El éxito o el fracaso de un sitio web orientado al usuario común vendrá determinado, entre otros aspectos, por el dimensionamiento del sistema sobre el que se instala y soporta el software que sustenta dicho sitio. En consecuencia, uno de los requisitos fundamentales de una aplicación web es que sea completamente escalable sin que un aumento de los recursos dedicados a la misma suponga modificación alguna en su comportamiento o capacidades.

Escalabilidad de un sistema web

La escalabilidad de un sistema web puede ser:

- **Verticalmente:** de manera ascendente "upgrades" a cada nodo.
- **Horizontalmente:** consiste en aumentar el número de nodos.
- **Cluster:** consiste en crear agrupaciones de servidores.

Escalabilidad vertical

Habitualmente, la separación lógica en capas se implementa de tal forma que se permita una separación física de las mismas. Interponiendo elementos conectores que actúen de "middlewares" es posible distribuir la aplicación de forma vertical (una máquina por cada capa del sistema), e incluso si esto no fuera suficiente, distribuyendo los elementos de una misma capa entre distintas máquinas servidoras.

Escalabilidad horizontal

Se trata de clonar el sistema en otra máquina de características similares y balancear la carga de trabajo mediante un dispositivo externo. El balanceador de carga puede ser:

- **Balanceador Software:** Por ejemplo, es común encontrar un servidor web Apache junto con el módulo `mod_jk`, que permite redirigir las solicitudes HTTP configuradas entre las diferentes máquinas que conforman la granja de servidores. Estos balanceadores examinan el paquete HTTP e identifican la sesión del usuario, registrando qué máquina de la granja está atendiendo dicha sesión. Esto es crucial, ya que nos permite diseñar la aplicación teniendo en cuenta el objeto de sesión del usuario y almacenar información relevante de la sesión del mismo. Con esta garantía, todas las peticiones de una misma sesión HTTP se redirigen a la misma máquina.
- **Balanceador hardware:** Son dispositivos que, siguiendo algoritmos de reparto de carga (como Round Robin, LRU - Menos Usado Recientemente, etc.), redirigen una solicitud HTTP del usuario a la máquina que corresponde según dicho algoritmo. Estos son más rápidos que los anteriores, ya que se basan en conmutación de circuitos y no examinan ni interpretan el paquete HTTP. Sin embargo, no aseguran el mantenimiento de la misma

sesión de usuario en la misma máquina. Esto condiciona el diseño, ya que obliga a almacenar la información de sesión del usuario por parte del desarrollador, en cookies o en una base de datos.

- **Balanceador hardware HTTP:** Son dispositivos hardware que examinan el paquete HTTP y mantienen la relación entre el usuario y la máquina servidora. Son más rápidos que los balanceadores software, pero ligeramente menos que los balanceadores hardware. Hoy en día, representan una de las soluciones más aceptadas en el mercado.

Cluster:

Con la introducción de servidores de aplicaciones en cluster se abre una nueva capacidad de escalabilidad que puede clasificarse como vertical u horizontal, dependiendo de su implementación. Un cluster de servidores de aplicaciones permite desplegar una aplicación web convencional, distribuyendo su carga de trabajo entre la granja de servidores del cluster. Esto ocurre de manera transparente tanto para el usuario como para el administrador. Mediante el mecanismo de replicación de sesión, el cluster garantiza que sin importar cuál máquina atienda la solicitud HTTP, esta tendrá acceso a la sesión del usuario (objeto HttpSession en Java). Sin embargo, debido a la replicación de sesión, este tipo de sistemas suele presentar problemas de rendimiento.

Referencias

IOC - Institut obert de Catalunya

https://apuntes-daw.javiergutierrez.trade/despliegue-de-aplicaciones/ut1/recopila.html