PRACTICA ARQUITECTURAS WEB (corregido)

Aspectos generales de la Arquitectura Web

La arquitectura de un sitio web es la forma en que se estructuran, jerarquizan y enlazan las urls y los contenidos entre sí.

Esta planificación debe definirse (preferiblemente) antes de desarrollar el proyecto para evitar reestructuraciones y correcciones.

Una aplicación Web es proporcionada por un servidor Web y utilizada por usuarios que.se Conectan desde cualquier punto vía clientes Web (browsers o navegadores). Por lo tanto, tienen una arquitectura **Cliente/servidor**.

La arquitectura de un Sitio Web tiene tres componentes principales:

- Un servidor Web
- Una conexión de red
- Uno o más clientes

El **servidor Web** distribuye páginas de información formateada a los clientes que las solicitan. Los requerimientos son hechos a través de una conexión de red, y para ello **se usa el protocolo HTTP.** Una vez que se solicita esta petición mediante el protocolo HTTP y la recibe el servidor Web, éste localiza la página Web en su sistema de archivos y la envía de vuelta al navegador que la solicitó.

Una vez que se entrega una página, la conexión entre el browser y el servidor Web se rompe, es decir que la lógica del negocio en el servidor solamente se activa por la ejecución de los scripts de las páginas solicitadas por el browser (en el servidor, no en el cliente).

La colección de páginas son en una buena parte dinámicas (ASP, PHP, etc.), y están agrupadas lógicamente para dar un servicio al usuario. El acceso a las páginas está agrupado también en el tiempo (sesión). Los componentes de una aplicación Web son:

1. Lógica de negocio.
    - Parte más importante de la aplicación.
    - Define los procesos que involucran a la aplicación.
    - Conjunto de operaciones requeridas para proveer el servicio.
2. Administración de los datos.
    - Manipulación de BD y archivos.
3. Interfaz
    - Los usuarios acceden a través de navegadores, móviles, PDAs, etc.
    - Funcionalidad accesible a través del navegador.
    - Limitada y dirigida por la aplicación.

Las aplicaciones web se modelan mediante lo que se conoce como modelo de capas, Una capa representa un elemento que procesa o trata información. Los tipos son:

- Modelo de dos capas: La información atraviesa dos capas entre la interfaz yla administración de los datos.
- Modelo de n-capas: La información atraviesa varias capas, el más habitual es el modelo de tres capas.

MODELO DE DOS CAPAS

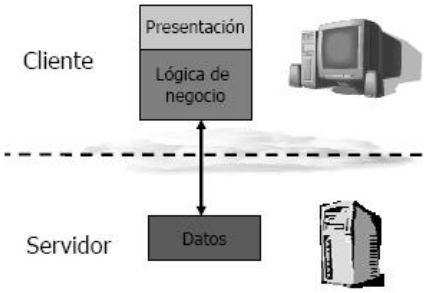
Gran parte de la aplicación corre en el lado del cliente. Se sigue utilizando mucho al día de hoy, Sin un modelo como este no existiría lo que hoy se conoce navegar por la red, ya que el modelo permite "cumplir" una petición de un cliente a uno o varios servidores y este mostrar los resultados de la petición

Las capas son:

1. **Cliente (fat client):** La lógica de negocio está inmersa dentro de la aplicación que realiza el interfaz de usuario, en el lado del cliente.
2. **Servidor:** Este nivel de la Base de Datos también llamado el Repositorio de Datos, es la capa en donde se almacena toda la información ingresada en el sistema y que se deposita en forma permanente.

Las limitaciones de este modelo son.

- Es difícilmente escalable
- Número de conexiones reducida
- Alta carga de la red.
- La flexibilidad es restringida
- La funcionalidad es limitada.



Los beneficios de este modelo son:

Un ejemplo simple: podría ser el funcionamiento de un juego online. Si existen dos servidores de juego, cuando un usuario lo descarga y lo instala en su computadora pasa a ser un cliente. Si tres personas juegan en un solo computador existirían dos servidores, un cliente y tres usuarios. Si cada usuario instala el juego en su propio ordenador existirían dos servidores, tres clientes y tres usuarios.

- Permite acceder a información clara y trasparente.
- Permite comunicación entre uno o más personas.
- Permite acceso/obtención de archivos voz y datos.
- Permite conocer el mundo empresarial.

MODELO DE TRES CAPAS:

Entre presentación y los datos, los procesos pueden ser manejados de forma separada a la interfaz de usuario y a los datos, esta capa intermedia centraliza la lógica de negocio, haciendo la administración más sencilla, los datos se pueden integrar de múltiples fuentes, las aplicaciones web actuales se ajustan a este modelo.

Las capas de este modelo son:

1. Capa de presentación (parte en el cliente y parte en el servidor)
    - Recoge la información del usuario y la envía al servidor (cliente)
    - Manda información a la capa de proceso para su procesado
    - Recibe los resultados de la capa de proceso
    - Generan la presentación
    - Visualizan la presentación al usuario (cliente)

2. Capa de proceso (servidor web)
    - Recibe la entrada de datos de la capa de presentación
    - Interactúa con la capa de datos para realizar operaciones
    - Manda los resultados procesados a la capa de presentación
3. Capa de datos (servidor de datos)
    - Almacena los datos
    - Recupera datos
    - Mantiene los datos
    - Asegura la integridad de los datos

Evolución de los servicios Web

Un servicio Web facilita un servicio a través de Internet: se trata de una interfaz mediante la que dos máquinas (o aplicaciones) se comunican entre sí. Esta tecnología se caracteriza por estos dos rasgos:

- **Multiplataforma:** cliente y servidor no tienen por qué contar con la misma configuración para comunicarse. El servicio web se encarga de hacerlo posible.
- **Distribuida:** por lo general, un servicio web no está disponible para un único cliente, sino que son diferentes los que acceden a él a través de Internet.

La evolución de la Web, básicamente, ha sido la siguiente:

- **Web 1.0**: la primera que apareció hacia 1990. En ella solo se podía consumir contenido. Se trataba de información a la que se podía acceder, pero sin posibilidad de interactuar; era unidireccional.
- **Web 2.0**: apareció en 2004 y contiene los foros, los blogs, los comentarios y después las redes sociales. La web 2.0 permite compartir información. Y aquí estamos, de momento la mayor parte de los consumidores.
- **La web 3.0:** fue operativa en el 2010 y se asocia a la web semántica, un concepto que se refiere al uso de un lenguaje en la red. Por ejemplo, la búsqueda de contenidos utilizando palabras clave.
- **Web 4.0**: empezó en el 2016 y se centra en ofrecer un comportamiento más inteligente y más predictivo, de modo que podamos, con sólo realizar una afirmación o una llamada, poner en marcha un conjunto de acciones que tendrán como resultando aquello que pedimos, deseamos o decimos

Tecnologías asociadas a las aplicaciones Web

En las tecnologías asociadas a las aplicaciones Web debemos distinguir entre:

- Las tecnologías que se ejecutan en el lado del cliente
- Las tecnologías asociadas en el lado del servidor:

Tecnologías asociadas al lado del cliente:

Las tecnologías Web asociadas al lado del cliente, o front-end, son aquellas que corren en el navegador del usuario y que son básicamente tres: HTML, CSS y JavaScript. El frontend se enfoca en el usuario, en todo con lo que puede interactuar y lo que ve mientras navega.

HTML: HTML es el lenguaje de marcado estándar que se usa para crear páginas y aplicaciones web. Sus elementos forman los bloques de creación de las páginas y representan texto con formato, imágenes, entradas de formulario y otras estructuras. Cuando un explorador realiza una solicitud a una dirección URL, **con independencia de que se obtenga una página o una aplicación**, lo primero que se devuelve es un documento HTML.

CSS: Se usa para controlar la apariencia y el diseño de los elementos HTML.

JavaScript: Es el lenguaje de programación de la web. Es un lenguaje de programación interpretado y dinámico. Se utiliza para realizar acciones como:

- Seleccionar un elemento HTML y recuperar o actualizar su valor.
- Consultar datos en una API web.
- Enviar un comando a una API web (y responder a una devolución de llamada con su resultado).
- Realizar la validación.

Tecnologías asociadas al lado del cliente

Es importante destacar que los lenguajes de programación del lado del servidor son necesarios porque para hacer la mayoría de las aplicaciones web se debe tener acceso a muchos recursos externos a la computadora del cliente, principalmente bases de datos alojadas en servidores de Internet**. *Un caso claro es un banco***: no tiene ningún sentido que el cliente tenga acceso a toda la base de datos, sólo a la información que le concierne.

Para escribir páginas dinámicas de servidor existen varios lenguajes:

- **Practical Extraction and Report Language (PERL):** La ventaja más importante de PERL sobre C es que PERL no necesita ser recompilado, es un lenguaje interpretado.
- **Active Server Pages (ASP):** es una tecnología del lado servidor de Microsoft para páginas web generadas dinámicamente, que ha sido comercializada como un anexo a Internet Information Server (IIS).
- **Java Server Pages (JSP):** es la tecnología para generar páginas web de forma dinámica en el servidor, desarrollado por Sun Microsystems, basado en scripts que utilizan una variante del lenguaje Java.
- **Hipertext Preprocesor (PHP):** PHP (acrónimo recursivo de «PHP: Hypertext Preprocessor», originado inicialmente del nombre PHP Tools, o Personal Home Page Tools) es un lenguaje de programación interpretado. Se utiliza entre otras cosas para la programación de páginas web activas, y se destaca por su capacidad de mezclarse con el código HTML.

Tipos de aplicaciones web

En función de cómo presentan el contenido a mostrar:

Aplicación Web estática. Este tipo de web app muestran poca información y no suele cambiar mucho. Modificar los contenidos de las apps estáticas no es sencillo. Para hacerlo es necesario editar el HTML y actualizarlo en el servidor. Estos cambios serán, normalmente, responsabilidad del webmaster o de la empresa de desarrollo que programó el diseño de la web app. Un portfolio, un curriculum digital o una página de presentación de empresa serían ejemplos de aplicaciones web estáticas.

Aplicación Web dinámica. Utilizan bases de datos para cargar a información y estos contenidos se actualizan cada vez que el usuario accede a la web app. El proceso de actualización es muy sencillo y ni siquiera necesita entrar en el servidor para modificarlo. Además permite implementar muchas funcionalidades, como foros o bases de datos. Las aplicaciones web dinámicas suelen contar con un panel de administrador (CMS) para realizar cambios.

Tienda virtual o e-commerce. El desarrollo es más complicado porque debe permitir pagos electrónicos a través de tarjeta de crédito, PayPal u otro método de pago. El desarrollador también deberá crear un panel de gestión para el administrador. A partir de él se subirán, actualizarán o eliminarán los productos y se podrán gestionar pedidos y los pagos. En este caso la web app se ajusta al dispositivo móvil como una aplicación móvil, permitiendo interactuar con ella como si fuera una app nativa

Portal Web. Con portal nos referimos a un tipo de aplicación en el que la página principal permite el acceso a diversos apartados, categorías o secciones. Algunos ejemplos:

- Foros
- Chats
- Correo electrónico
- Buscadores
- Zona de acceso con registro

Aplicación Web con gestor de contenidos. En el caso de aplicaciones web en las que el contenido se debe ir actualizando continuamente lo mejor es recurrir a un un gestor de contenidos (CMS) a través del cual el administrador pude ir realizando los cambios y actualizaciones él mismo. Estos gestores son intuitivos y muy sencillos de gestionar.

Este tipo de aplicación web es muy común para páginas de contenidos, como blogs, páginas de noticias o medios de comunicación.

Los gestores de contenidos más utilizados son:

- **WordPress:** Sin duda es el más extendido de los gestores de contenidos. Existe mucha información en la red, tutoriales y guías para personalizarlo, entenderlo y además es gratuito.
- **Joomla:** Es el segundo en el top CMS, tras WordPress. Aunque no goza de tantos usuarios sí que tiene una comunidad potente.
- **Drupal:** Es un CSM de software libre. Es muy adaptable, y recomendado especialmente para generar comunidades.

Arquitecturas web. Modelos

Los modelos de arquitecturas web describen la relación entre los distintos elementos que componen la estructura de funcionamiento de las páginas y aplicaciones web. Los diferentes modelos que podemos encontrar en la actualidad son:

- Modelo Punto a Punto
- Modelo Cliente-Servidor
- Modelo con servidor de aplicaciones
- Modelo con servidor de aplicaciones externo
- Modelo con varios servidores de aplicaciones

Modelo Punto a Punto (P2P - Peer to Peer). Es un tipo de arquitectura de red descentralizada y distribuida en la que los nodos individuales de la red (peers) actúan tanto como suministradores como clientes de recursos.

En una red P2P, las tareas, como realizar un streaming de audio o vídeo, se comparten entre múltiples puntos interconectados que ponen una parte de sus recursos (CPU, almacenamiento, ancho de banda) disponibles directamente por otros puntos de la red, sin la necesidad de disponer de servidores que realicen la coordinación centralizada.

Ventajas

- **Escalabilidad:** Millones de usuarios potenciales. Cuantos más nodos conectados, mejor será su funcionamiento, al contrario del modelo C/S.
- **Robustez:** Los clientes pueden localizar los recursos sin depender del correcto funcionamiento de un único servidor.
- **Descentralización:** Ningún nodo es imprescindible para el funcionamiento de la red.
- **Distribución de costes entre los nodos:** No hace la carga sobre un único nodo de la red.

Desventajas

- **Falta de fiabilidad de los recursos:** No hay garantías de que los recursos obtenidos desde un nodo sean realmente los deseados. En el modelo C/S, el recurso obtenido del servidor central es único, y no ha podido ser modificado por otro nodo.
- **Difícil mantenimiento:** La actualización de los recursos compartidos debe realizarse en todos los nodos.

El modelo cliente-servidor es un modelo informático que actúa como una aplicación distribuida que particiona tareas o cargas de trabajo entre los proveedores de un recurso o servicio, llamados servidores, y los solicitantes del servicio, llamados clientes.

El esquema de funcionamiento más básico del modelo cliente-servidor para una arquitectura web está basado en uno o varios clientes que solicitan una página web a un servidor web:

- Desde el navegador web (cliente) el usuario solicita la carga de una página web indicando su URL.
- El servidor recibe la petición de la página web
- Busca en sus sistema de almacenamiento la página solicitada.
- Envía el contenido de la página web (código fuente) por el mismo medio por el que recibió la petición.
- El navegador web recibe el código fuente de la página y lo interpreta mostrando al usuario la página web.

El servidor web envía al cliente el recurso solicitado sin hacer ningún tratamiento sobre él.

Ventajas

- **Centralización del control:** los accesos, recursos y la integridad de los datos son controlados por el servidor de forma que un programa cliente defectuoso o no autorizado no pueda dañar el sistema. Esta centralización también facilita la tarea de actualizar datos u otros recursos (mejor que en las redes P2P)..
- **Escalabilidad:** se puede aumentar la capacidad de clientes y servidores por separado. Cualquier elemento puede ser aumentado (o mejorado) en cualquier momento, o se pueden añadir nuevos nodos a la red (clientes y/o servidores).
- **Fácil mantenimiento:** al estar distribuidas las funciones y responsabilidades entre varios ordenadores independientes, es posible reemplazar, reparar, actualizar, o incluso trasladar un servidor, mientras que sus clientes no se verán afectados por ese cambio (o se afectarán mínimamente). Esta independencia de los cambios también se conoce como encapsulación.
- **Existen tecnologías suficientemente desarrolladas,** diseñadas para el paradigma de C/S que aseguran la seguridad en las transacciones, la amigabilidad de la interfaz, y la facilidad de empleo.

Desventajas

La congestión del tráfico. Cuando una gran cantidad de clientes envían peticiones simultaneas al mismo servidor, puede ser que cause muchos problemas para éste (a mayor número de clientes, más problemas para el servidor). Al contrario, en las redes P2P como cada nodo en la red hace también de servidor, cuanto más nodos hay, mejor es el ancho de banda que se tiene.

- **No tiene la robustez de una red P2P**. Cuando un servidor está caído, las peticiones de los clientes no pueden ser satisfechas. En la mayor parte de redes P2P, los recursos están generalmente distribuidos en varios nodos de la red. Aunque algunos salgan o abandonen la descarga; otros pueden todavía acabar de descargar consiguiendo datos del resto de los nodos en la red.
- **El software y el hardware de un servidor son generalmente muy determinantes**. Un hardware regular de un ordenador personal puede no poder servir a cierta cantidad de clientes. Normalmente se necesita software y hardware específico, sobre todo en el lado del servidor, para satisfacer el trabajo. Por supuesto, esto aumentará el coste.
- **El cliente no dispone de los recursos que puedan existir en el servidor.** Por ejemplo, si la aplicación es una Web, no podemos escribir en el disco duro del cliente o imprimir directamente sobre las impresoras sin sacar antes la ventana previa de impresión de los navegadores.

En el modelo con servidor de aplicaciones, La función que realiza un servidor de aplicaciones es diferente, ya que los recursos que va a manipular no son archivos estáticos, sino que contienen el código que tiene que ejecutar. Es decir, un servidor web en solitario eviaría al cliente el recurso solicitado tal cual, mientras que el servidor de aplicaciones lo ejecuta y envía al cliente el resultado a través del servidor web.

Es muy frecuente que el servidor de aplicaciones deba conectarse con una base de datos para obtener los datos solicitados durante la ejecución del código. Dicha base de datos puede residir en la misma máquina que el servidor web o en otro host conectado en red.

En el modelo con servidor de aplicaciones externo, la parte correspondiente al servidor web suele tener menos carga de trabajo que el servidor de aplicaciones por lo que se puede establecer una estructura en la que el servidor web sea independiente del servidor de aplicaciones, de manera que el servidor web se pueda implantar de forma dedicada precisando menos recursos. Un redirector

será el encargado de transferir al servidor de aplicaciones los elementos que necesiten ser ejecutados, mientras que no pasarán del servidor web los recursos estáticos.

En el modelo con varios servidores de aplicaciones, si la carga de trabajo del servidor de aplicaciones se estima que va a ser elevado, se puede implantar un sistema con varios servidores de aplicaciones unidos a un mismo servidor web que requiere menos rendimiento. La conexión se realizará a través de un redirector como en el caso anterior, que además realizará las funciones de balanceador de carga para determinar en un determinado momento qué servidor de aplicaciones debe ejecutar el código en función de la carga de trabajo que tenga cada uno. En este caso, todos los servidores de aplicaciones deben ser iguales.

Plataformas web libres y propietarias

Las plataformas Web propietarias son plataformas por las que hay que pagar para tener derecho a la instalación y al mantenimiento, este valor varía dependiendo del número de usuarios. Por lo general, el pago es anual y para la renovación del contrato de la licencia de la plataforma se debe pagar de nuevo.

Las plataformas propietarias incluyen herramientas y aplicaciones muy completas y complejas que permiten una mayor facilidad en el seguimiento de un curso virtual.

Las plataformas Web libres son plataformas por las que no hay que pagar para tener derecho a su instalación y mantenimiento.

En ambas plataformas podemos encontrar:

- **Código abierto**. Es una filosofía de trabajo y colaboración seguida por los miembros de la comunidad de open source. Esta filosofía se basa en la libertad intelectual y los principios fundamentales: transparencia, colaboración, entrega, inclusión y comunidad. El intercambio de ideas y software desarrollado por las comunidades ha impulsado el avance creativo, científico y tecnológico en industrias tales como: educación, gobierno, derecho, salud y manufactura. Este movimiento creó una instancia para que los miembros de la comunidad global colaboren, compartan y se ayuden entre sí para lograr objetivos personales y comerciales a través del código fuente.
- **Código cerrado.** Los plataformas distribuidos como software de código cerrado generalmente solo incluyen los archivos necesarios para ejecutar la aplicación. No se incluyen los programas fuentes para modificar o adaptar el software a las necesidades del usuario.