TEMA 1. IMPLANTACIÓN DE ARQUITECTURAS WEB

Índice de contenido

Introducción: ¿Qué es un servicio Web?

Existen múltiples definiciones sobre lo que son los **servicios web**, lo que muestra su complejidad a la hora de dar una adecuada definición que englobe todo lo que son e implican.
Podemos considerar los servicios web como un **conjunto de aplicaciones o de tecnologías con capacidad para interoperar en la Web**. Estas aplicaciones o tecnologías intercambian datos entre sí con el objetivo de ofrecer unos servicios.

Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web.

Estos servicios proporcionan mecanismos de comunicación estándares entre diferentes aplicaciones, que interactúan entre sí para presentar información dinámica al usuario. Para proporcionar interoperabilidad y extensibilidad entre estas aplicaciones, y que al mismo tiempo sea posible su combinación para realizar operaciones complejas, es necesaria una arquitectura de referencia estándar.

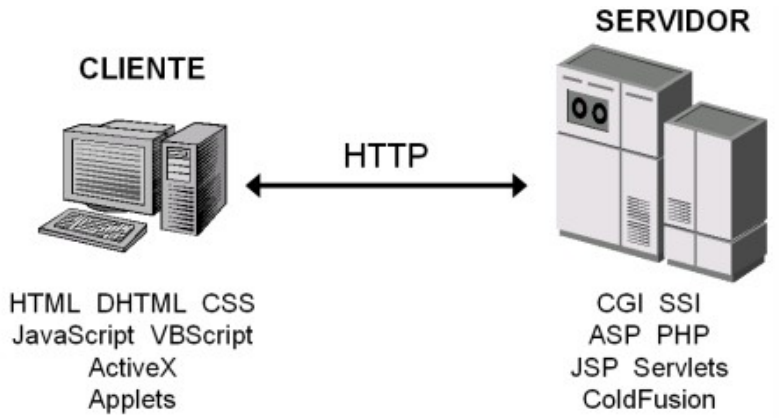
Definición: Un **servicio o aplicación web** es un **programa que sirve para atender y responder a las diferentes peticiones de los navegadores, proporcionando los recursos que soliciten. Esto se lleva a cabo en un entorno web usando arquitectura cliente-servidor y siguiendo unas reglas de comunicación establecidas por el protocolo HTTP o el protocolo HTTPS** (la versión cifrada y autenticada).

Arquitectura Cliente-Servidor

Es la configuración arquitectónica más habitual usada en servicios web. Se basa en la idea de servicio, en el que el cliente es un componente consumidor de servicios y el servidor es un proceso proveedor de servicios. Además, esta relación está robustamente cimentada en **el intercambio de mensajes** como el único elemento de acoplamiento entre ambos.

La arquitectura cliente-servidor aplicada a los servicios web, tiene los siguientes actores:

– Cliente: un navegador
– Servidor: un servidor web
– Comunicación: protocolo HTTP o HTTPS.



Arquitectura TCP/IP

Esta arquitectura de red es la base para las comunicaciones en Internet. La arquitectura TCP/IP se desarrolló ligada a los sistemas operativos UNIX y Linux. Aunque actualmente, debido a su gran utilización, también se ha implantado en otros sistemas operativos como Windows. Surge en 1973 dentro de un proyecto del departamento de defensa americano (DoD), cuyo objetivo era desarrollar una red, Arpanet, con las siguientes características:

- Permitir la interconexión de redes diferentes
- Ser tolerante a fallos
- Proporcionar el uso de diferentes aplicaciones.

Para alcanzar este objetivo se diseña una red con las siguientes características:

- Topología irregular o mixta.
- Fragmentación de la información para seguir diferentes rutas hacia el destinatario
- Si alguna de las rutas falla, la información podrá seguir una ruta alternativa.

Su éxito reside en que es capaz de interconectar redes de diferentes tecnologías y fabricantes, funciona sobre máquinas de cualquier tamaño (debido a que no necesita muchos recursos), soporta múltiples topologías de redes y, no es una solución propietaria de un fabricante. Esto ha hecho que la arquitectura TCP/IP se convierta en un estándar de facto.

La arquitectura del World-Wide Web

El diseño del World-Wide Web sigue el modelo cliente-servidor. En la WWW, nuestras estaciones de trabajo son clientes que demandan hipertextos a los servidores. Para poner en marcha un sistema como éste ha sido necesario:

a) Diseñar e implementar un nuevo protocolo que permitiera realizar saltos hipertextuales, esto es, de un nodo de origen a uno de destino, que podria ser un texto o parte de un texto, una imagen, un sonido, una animación, fragmento de vídeo, etc. Es decir, cualquier tipo de información en formato electrónico. Este protololo se denomina HTTP (HyperText Transfer Protocol) y es el "lenguaje" que "hablan" los servidores del WWW.

b) Inventar un lenguaje para representar hipertextos que incluyera información sobre la estructura y el formato de representación y, especialmente, indicar origen y destino de saltos hipertextuales. Este lenguaje es el HTML o (HyperTextex Markup Language).

c) Idear una forma de codificar las instrucciones para los saltos hipertextuales de un objeto a otro de la Internet. Dada la variedad de protocolos, y por tanto, formas de almacenamiento y recuperación de la información, en uso en la Internet, esta información es vital para que los clientes (ver el siguiente punto) puedan acceder a dicha información.

d) Desarrollar aplicaciones cliente para todo tipo de plataforma y resolver el problema de cómo acceder a información que está almacenada y es accesible a través de protocolos diversos (FTP, NNTP, Gopher, HTTP, X.500, WAIS, etc.) y representar información multiformato (texto, gráficos, sonidos, fragmentos de vídeo, etc.).

HTTP (HyperText Transfer Protocol)

El HTTP (HyperText Transfer Protocol) es el protocolo de alto nivel del World-Wide Web que rige el intercambio de mensajes entre clientes y servidores del Web. Recordemos que un protocolo es descripción formal de los formatos de los mensajes y las reglas que deben seguir dos ordenadores para intercambiar dichos mensajes

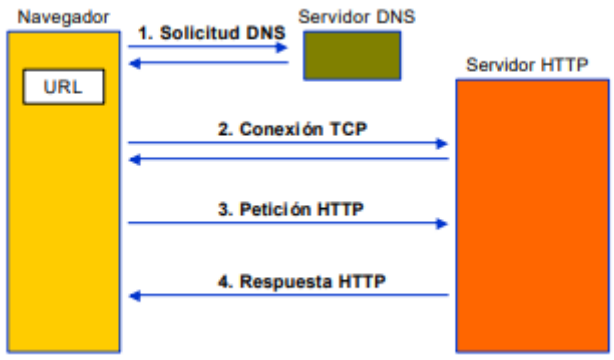
El HTTP es un protocolo genérico orientado a objetos que no mantiene la conexión entre transacciones (Tim Berners-Lee, 1993d). Sus características principales son:

- Ligereza: reduce la comunicación entre clientes y servidores a intercambios discretos, de modo que no sobrecarga la red y permite saltos hipertextuales rápidos.
- Generalidad: puede utilizarse para transferir cualquier tipo de datos, según el estándar MIME **[5]**. El cliente y el servidor pueden negociar en cualquier momento el modo de representación de los datos: el cliente notifica al servidor una lista de formatos que entiende, y en adelante el servidor sólo remitirá al cliente datos que este sea capaz de manejar. El cliente debe aceptar al menos dos formatos: text/plain (texto normal) y text/html (hipertexto codificado en HTML.
- Extensibilidad: contempla distintos tipos de transacción entre clientes y servidores ("métodos", en la jerga HTTP), y la futura implementación de otros nuevos. Esto abre posibilidades más allá de la simple recuperación de objetos de la red: búsquedas, anotaciones, etc.

El esquema básico de cualquier transacción HTTP entre un cliente y un servidor se muestra a continuación (Tim Berners-Lee):

Funcionamiento básico de petición de servicio web (Transferencia páginas web)

1. Traducción de nombre a IP. Solicitud su servidor DNS.
2. El servidor espera peticiones en el puerto TCP.
   - El cliente (navegador) envía la petición (solicita una URL)
   - Apertura de la conexión con el servidor
3. El servidor hace búsqueda de la página o recurso.
4. Envío (o código de error) por parte del servidor (por la misma conexión por la que acepto la solicitud)
   - Interpretación del documento HTML y petición de otros objetos a los que hace referencia
   - Cierre conexión una vez que hayan llegado todos los recursos y documentos



Un servidor web que siga el esquema anterior cumplirá todos los requisitos básicos de los servidores HTTP, aunque solo podrá servir ficheros estáticos.

A partir del anterior esquema se han diseñado y desarrollado todos los servidores de HTTP que existen, variando solo el tipo de peticiones (páginas estáticas, CGI, Servlets, etc.)

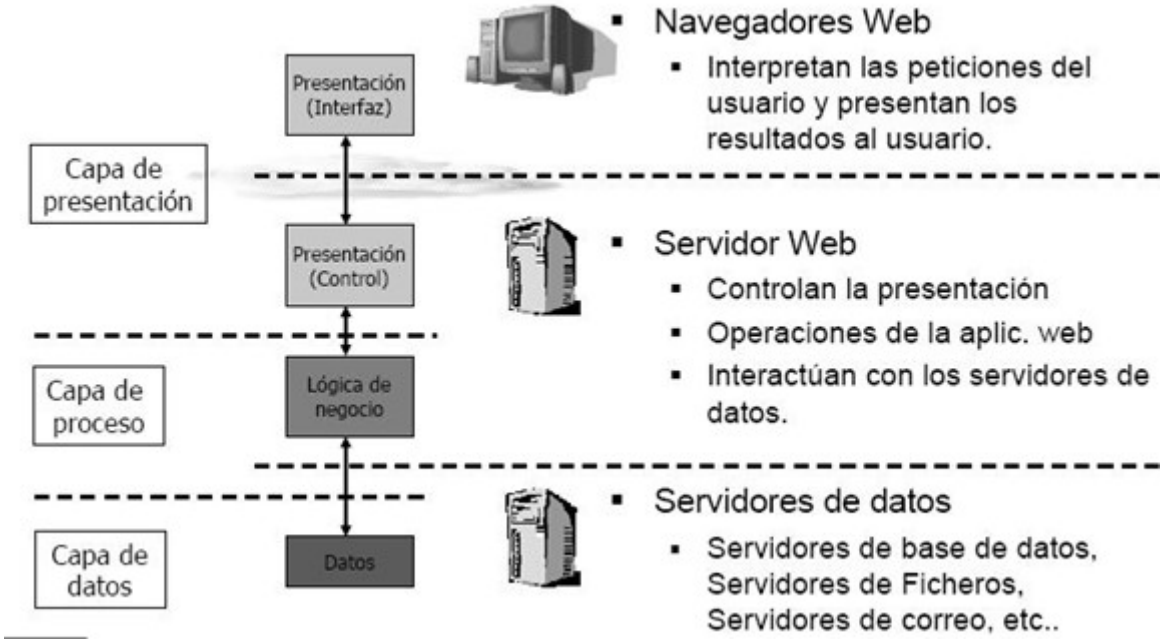
1.- Aspectos generales de arquitecturas web.

La arquitectura World Wide Web (WWW) de Internet provee un modelo de programación sumamente poderoso y flexible. Las aplicaciones y los contenidos son presentados en formatos de datos estándar y son localizados por aplicaciones conocidas como “web browsers”, que envían requerimientos de objetos a un servidor y éste responde con el dato codificado según un formato estándar. Los estándares WWW especifican muchos de los mecanismos necesarios para construir un ambiente de aplicación de propósito general, por ejemplo:

- **Modelo estándar de nombres**: todos los servidores, así como el contenido de la WWW se denominan según un Localizador Uniforme de Recursos (Uniform Resource Locator: URL).
- **Contenido**: a todos los contenidos en la WWW se les especifica un determinado tipo permitiendo de esta forma que los browsers (navegadores) los interpreten correctamente.
- **Formatos de contenidos estándar**: todos los navegadores soportan un conjunto de formatos estándar, por ejemplo HTML, XML, ECMA, JavaScript, etc.
- **Protocolos estándar**: éstos permiten que cualquier navegador pueda comunicarse con cualquier servidor web. El más comúnmente usado en WWW es HTTP (Protocolo de Transporte de Hiper-Texto), que opera sobre el conjunto de protocolos TCP/IP.

Esta infraestructura permite a los usuarios acceder a una gran cantidad de aplicaciones y servicios de terceros. También permite a los desarrolladores crear aplicaciones y servicios para una gran comunidad de clientes.

De forma genérica podríamos decir que la **arquitectura web es un modelo compuesto de tres capas**:



1. **Capa de Base de Datos**, donde estaría toda la documentación de la información que se pretende administrar mediante el servicio web y emplearía una plataforma del tipo MySQL, PostgreSQL, etc.
2. En una segunda capa estarían los **servidores de aplicaciones web**, ejecutando aplicaciones de tipo Apache, Tomcat, Resin, etc.
3. En una tercera capa estarían los **clientes del servicio web** al que accederían mediante un navegador web como Chrome, Firefox, Internet Explorer, Opera, etc.

1.1.- Evolución de los servicios/aplicaciones web.

La evolución del uso de Servicios web en las organizaciones está fuertemente ligada al desarrollo de Internet como red prestadora de servicios. Entre los factores que han impulsado el uso de los servicios web se encuentran:

- El **contenido se está volviendo más dinámico**: Los sitios web actuales proporcionan contenidos "instantáneos". Un Servicio web debe ser capaz de combinar contenido proveniente de fuentes muy diferentes.
- El **ancho de banda es menos costoso**: Actualmente un Servicio web puede entregar tipos variables de contenidos como vídeo o audio. A medida que crezca el ancho de banda, los servicios web deben adaptarse a nuevos tipos de contenidos.
- El **almacenamiento es más barato**: Un Servicio web debe ser capaz de manejar cantidades masivas de datos, y debe poder hacerlo de forma inteligente.
- El **éxito de la computación extendida** se está volviendo más importante: Con cientos de millones de dispositivos como teléfonos móviles, agendas electrónicas, etc. existentes actualmente, estamos llegando a un momento en el cual las computadoras están dejando de ser el dispositivo más común en Internet. A medida que las plataformas se hacen más diversas, tecnologías como XML se hacen más importantes. Un servicio web no puede exigir que los usuarios ejecuten, por ejemplo, un navegador web tradicional en alguna versión de Microsoft Windows; por el contrario, los servicios web deben servir a todo tipo de dispositivos, plataformas y navegadores, entregando contendido sobre una amplia variedad de tipos de conexión.

Estos factores, unidos a los beneficios proporcionados por los servicios web en la organización y los buenos productos disponibles para su desarrollo, han hecho que su utilización se extienda sin mayores obstáculos.

En los orígenes del mundo web (Web 1.0) nos situábamos ante un entorno estático, con páginas en formato HTML que raramente sufrían modificaciones o actualizaciones y en las que apenas había interacción con el usuario.

Con la **Web 2.0** , empezamos a usar unas webs colaborativas donde los usuarios dejan de ser totalmente pasivos y comienzan a aportar conocimientos, comentarios, etc. Es la etapa de las redes sociales y las wikis. Además, aplicaciones tradicionales que están fuertemente enfocadas al usuario final como por ejemplo un procesador de texto, empiezan a funcionar a través de la web. En este nuevo entorno existen una serie de nuevas tecnologías que, en general, tienen como objetivo:

- Transformar software de escritorio hacia la web.

- Separar hojas de estilo.
- Potenciar el trabajo colaborativo y la utilización de redes sociales.
- Dar control total a los usuarios en el manejo de su información.

En la ya **Web 3.0**. introducimos el concepto de web semántica, intentando acercar el uso del lenguaje natural y donde se pretende que la navegación se adapte a los gustos del usuario.
Una definición podría ser:

> *"Un nuevo paradigma para la web que no sólo permite la conversación e interacción entre sus usuarios, sino que además permite actuar de forma proactiva y ayuda a los usuarios a realizar una navegación más personalizada"*

Esta Web 3.0. nace ante la necesidad de ayudar a los usuarios a encontrar la información que ese usuario concretamente necesita. Debido a la gran avalancha de información, trata de proporcionarle al usuario las experiencias más cercanas.
Implica también una evolución tecnológica, aplicando nuevos lenguajes, nuevas técnicas de búsqueda y de almacenamiento. Pretende normalizar la información que se guarda, de modo que las búsquedas se puedan realizar con un lenguaje cercano al natural.

En definitiva hace la web más inteligente teniendo en cuentas nuestros gustos, preferencias, hábitos e incluso el contexto. Esto implica que en algún lugar quedan regsitrados todos nuestros movimientos en la red….¿Dónde queda entonces nuestra privacidad? Interesante cuestión que deberíamos meditar.

1.2.- Tipos de aplicaciones web.

Cualquier proyecto que se quiera desarrollar en Internet, bien sea comercio electrónico, reservas de billetes de vuelo on-line, información meteorológica, registro de usuarios, simuladores de hipotecas, etc, conlleva el desarrollo de una aplicación web. En definitiva, una aplicación web es una plataforma orientada a automatizar los procesos de servicios que se quieran ofrecer a usuarios.

Establecer una clasificación de los tipos de aplicaciones web es una tarea compleja debido a la dificultad existente para poder establecer algún parámetro en función del cual establecer dicha clasificación, junto con la innumerable cantidad de aplicaciones existentes.

En función de cómo se presenta la aplicación web junto con el contenido que pretende mostrar, se ha establecido la siguiente clasificación:

- **Página web Estática**. Están implementadas en HTML y pueden mostrar en alguna parte de la página objetos en movimiento tales como banners, GIF animados, vídeos, etc.
- **Página web Animada.** Permite que una página web presente el contenido con ciertos efectos animados continuados. El uso de esta tecnología permite diseños más vanguardistas, modernos y creativos Se realizan con la tecnología FLASH (cada vez menos usada) o también utilizar una combinación de lenguajes (como LESS conjuntamente con aplicaciones Jquery)
- **Página web Dinámica.** Existen muchos lenguajes de programación que son la base para la mayoría de páginas web dinámicas. Los que destacamos aquí son los lenguajes PHP y ASP. Estos lenguajes permiten una perfecta estructuración del contenido. Por una parte crearíamos la estructura de las páginas web y por otra, almacenaríamos el contenido en determinados archivos. A partir de ahí, crearíamos el código de llamada, que insertaría el

contenido en la propia página web estructurada. Este es el principio básico que siguen los lenguajes de programación. A partir de aquí se desarrollan aplicaciones para poder gestionar el contenido a través de un panel de control.

- **Portal.** Es un sitio web que en su página principal permite el acceso a múltiples secciones que, por lo general, son foros, chats, cuentas de correo, buscador, acceso registrado para obtener ciertas ventajas, las últimas noticias de actualidad, etc.
- **Tienda virtual o comercio electrónico.** Sitio web que publica los productos de una tienda en Internet. Permite la compra on-line a través de tarjeta de crédito, domiciliación bancaria o transferencia bancaria en general. Ofrece al administrador un panel de gestión para poder subir los productos, actualizarlos, eliminarlos, etc.
- **Página web con "Gestor de Contenidos".** Se trata de un sitio web cuyo contenido se actualiza a través de un panel de gestión por parte del administrador del sitio. Este panel de gestión suele ser muy intuitivo y fácil de usar. En aquellas páginas web que requieran una actualización constante, se suele incorporar este panel de gestión para que la web pueda controlarse día a día por parte del cliente.

Cuando adquirimos un equipo informático nuevo, existen una serie de aplicaciones imprescindibles que es necesario instalar junto con los drivers de nuestro equipo para poder empezar a utilizarlo. Entre estas aplicaciones encontramos aplicaciones ofimáticas, antivirus, aplicaciones de mensajería, compresores, reproductores multimedia, etc.

¿En algún momento te has parado a pensar qué cantidad de aplicaciones web hay disponibles en Internet para sustituir a las que tienes pensado instalar en el equipo?

1.3.- 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:

- El **sistema operativo,** bajo el cual opera el servidor donde se hospedan las páginas web y que representa la base misma del funcionamiento del computador. En ocasiones limita la elección de otros componentes.

  Los sistemas server más habituales son:
  - *Microsoft Windows Server* (versiones 2003, *2008, 2012 , 2016,...*)
  - *GNU/Linux Server: las distribuciones RedHat*, *Ubuntu Server*, *CentOS*, *SuSE Linux Enterprise Server*, …)
  - *UNIX* (*IBM AIX*, *HP-UX*)
  - *Solaris* /*OpenSolaris*
  - *Apple OS X Server*

- El **servidor web** es 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.

  Algunos web-server-software más usados son: Apache, Internet Information Services (IIS), Lighttpd , LiteSpeed, NginX, etc.

  https://en.wikipedia.org/wiki/Comparison_of_web_server_software

- El **gestor de bases de datos** se encarga de almacenar sistemáticamente un conjunto de registros de datos relacionados para ser usados posteriormente. Ej Postgre SQL, SQL Server, MySQL, MariaDB, etc.

- Un **lenguaje de programación interpretado** que controla las aplicaciones de software que corren en el sitio web (PHP, ASP...).

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.

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.

LEMP sería similar pero con Nginx en lugar de Apache.

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.

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

1.4.- Tecnologías asociadas a las aplicaciones web.

Las aplicaciones web emplean páginas dinámicas, éstas se ejecutan en un servidor web y se muestran en el navegador de un equipo cliente que es el que ha realizado previamente la solicitud. Cuando una página web llega al navegador, es posible que también incluya algún programa o fragmento de código que se deba ejecutar. Ese código, normalmente en lenguaje JavaScript, lo ejecutará el propio navegador. Es por ello que en

este apartado nos entraremos en las tecnologías asociadas a las aplicaciones web que se ejecutarán tanto del lado del servidor como del cliente, especificando lo que corresponda en cada uno de los casos.

Cliente

• Gestiona las peticiones del usuario y la recepción de las páginas que provienen del servidor
• Interpreta los documentos HTML y sus recursos.

Las tecnologías más empleadas son:

- HTML (HyperText Markup Language)
- CSS (Cascading Style Sheets), DHTML documento XML
- Lenguaje de script (**#JavaScript**, VBScript, etc.) . El código se incrusta dentro del HTML.
- Control ActiveX
- Applets en Java (requiere instalar en el cliente una Máquina Virtual Java - JVM)
- Plug-ins: Macromedia Flash, Autodesk MapGuide, ...
- Virtual Reality Modeling Language (VRML)

Servidor

• Programa residente que espera peticiones: demonio (**daemon**) en Unix y **servicio** en Microsoft
• En la aplicación del servidor hay:

- Páginas estáticas (documentos HTML)
- Recursos multimedia (imágenes y documentos adicionales del sitio web)
- Scripts o programas de servidor que al ser invocados se ejecutan y dan como resultado una página HTML generada (pueden acceder a una BD). Los principales lenguajes de script del servidor son : PHP, CGI, JSP, ASP, ASP.net, Python, RUBY.

• Tecnologías de servidor:

- CGI: complejo y poco eficiente
- SSI: estándar de “macros” de servidor web
- ASP (Microsoft): Windows, Windows NT
- JSP y Servlets (Sun Microsystems): Windows, algunos Unix
- PHP (PHP.net): código fuente, binarios para Win32 y algunos Unix
- ColdFusion (Macromedia/allaire): Windows, Windows NT, Linux, Solaris, HP-UX

Describimos algunos de estos conceptos:

- **JavaScript**: Lenguaje que se interpreta y se ejecuta en el cliente. Útil para realizar tareas como mover imágenes por la pantalla, crear menús de navegación interactivos, utilizar algunos juegos, etc. En las páginas web suele preferirse JavaScript porque es aceptado por muchos más navegadores que VBScript (creado por Microsoft).
- **VBScript (Visual Basic Scripting)**: La respuesta de Microsoft a JavaScript. VBScript es una buena herramienta para cualquier sitio destinado a ser mostrado exclusivamente en el navegador Microsoft Internet Explorer. El código en VBScript puede, además, estar diseñado para su ejecución en el lado del cliente o en el del servidor, la diferencia es que un código que se ejecuta en el lado del servidor no es visible en el lado del cliente. Éste recibe los resultados, pero no el código.
- **CSS (Cascading Style Sheets):** Las “Hojas de Estilo en Cascada” se usan para formatear las páginas web; se trata de separar el contenido de un documento de su presentación. Cualquier cambio en el estilo marcado para un elemento en la CSS afectará a todas las páginas vinculadas a esa CSS.
- **Applets de Java:** que son pequeños programas interactivos que se encuentran incrustados en una página web y pueden funcionar con cualquier tipo de navegador: Explorer, Netscape, Ópera, etc. El código de un applet es ejecutado por el navegador mismo, usando lo que se llama la máquina virtual Java (JVM). Los applets

proporcionan funcionalidad a páginas de Internet que no puede ser satisfecha usando únicamente HTML.

- **CGI (Common Gateway Interface):** La “Interface Común de Entrada” es uno de los estándares más antiguos en Internet para trasladar información desde una página a un servidor web. Este estándar es utilizado para bases de datos, motores de búsqueda, formularios, generadores de email automático, foros, comercio electrónico, rotadores y mapas de imágenes, juegos en línea, etc.

  Las rutinas de CGI son habitualmente escritas en lenguajes interpretados como Perl o por lenguajes compilados como C.

- **ASP (Active Server Pages)**: Tecnología propietaria de Microsoft que **permite crear páginas web dinámicas en el servidor.** Desarrollada con el objetivo de sustituir a la tecnología CGI, ofrece una serie de características que facilitan la programación de aplicaciones web. Las páginas ASP suelen estar programadas en VBScript, aunque también se pueden programar en otros lenguajes, como JScript.

  Existen versiones de ASP para Unix y Linux, a pesar de que fue una tecnología desarrollada por Microsoft para la creación dinámica de páginas web ofrecida junto a su servidor IIS.

- **JSP (Java Script Pages).** Idem ASP pero de Sun Microsystems . Se programan en Java. Tienen la arquitectura de separación de la lógica de negocio de los datos. Incluso también servicio HTTP, servicio de datos y servicio de aplicaciones por separado.

- **Servlet** : programa Java que se ejecuta en la parte servidor, suele usarse Tomcat. Con similitud a JSP, de hecho una JSP se puede trasformar en un servlet y ser ejecutado por Tomcat también.

- **CSP (Caché Server Pages)** Tecnología propietaria de Intersystems que permite crear páginas web dinámicas en el servidor. Se diferencia de otras tecnologías similares como ASP y JSP en que la lógica de negocio reside junto con la lógica de datos en el sistema gestor de bases de datos.

- **PHP (Hypertext Preprocessor)**: Este lenguaje es, como ASP, ejecutado en el lado del servidor. PHP es similar a ASP y puede ser usado en circunstancias similares. Es muy eficiente, permitiendo el acceso a bases de datos empleando servidores como MySQL (potente gestor de bases de datos relacional, sencillo de usar e increíblemente rápido. También es uno de los motores de bases de datos más usados en Internet, la principal razón de esto es que se distribuye bajo la licencia GNU GPL para aplicaciones no comerciales) y, por lo tanto, suele utilizarse para crear páginas dinámicas complejas.

1.5.- 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 delmomento. 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 éste 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.

La escalabilidad de un sistema web puede ser:

- Verticalmente: actualizando y ampliando prestaciones de cada servidor.
- Horizontalmente: consiste en aumentar el número de servidores.
- Cluster: consiste en crear agrupaciones de servidores.

2.- Servidor web Apache.

Un servidor web es un programa que se ejecuta de forma continua en un ordenador (también se utiliza el término para referirse al ordenador que lo ejecuta), se mantiene a la espera de peticiones por parte de un cliente (un navegador de Internet) y contesta a estas peticiones de forma adecuada,sirviendo una página web que será mostrada en el navegador o mostrando el mensaje correspondiente si se detectó algún error.

Uno de los servidores web más populares del mercado y el más utilizado actualmente es Apache, de código abierto y gratuito, disponible para Windows y GNU/Linux, entre otros.

En cuanto a su arquitectura podemos destacar lo siguiente:

- Estructurado en módulos.
- Cada módulo contiene un conjunto de funciones relativas a un aspecto concreto del servidor.
- El archivo binario **httpd** contiene un conjunto de módulos que han sido compilados.
- La funcionalidad de estos módulos puede ser activada o desactivada al arrancar el servidor.
- Los módulos de Apache se pueden clasificar en tres categorías:
  - Módulos base: Se encargan de las funciones básicas.
  - Módulos multiproceso: Encargados de la unión de los puertos de la máquina, aceptando las peticiones y atendiéndolas.
  - Módulos adicionales: se encargan de añadir funcionalidad al servidor.

El servidor Apache se desarrolla dentro del proyecto HTTP Server (httpd) de la Apache SoftwareFoundation. La licencia de software, bajo la cual el software de la fundación Apache es distribuido, esuna parte distintiva de la historia de Apache HTTP Server y de la comunidad de código abierto.

La **Licencia Apache** permite la distribución de derivados de código abierto y cerrado a partir de su código fuente original.

Para Saber más
Esta web sirve como manual de referencia, guía de usuario, tutoriales prácticos, etc., sobre el servidor web Apache. Se trata de la web oficial de Apache Software Foundation.
http://httpd.apache.org/docs/2.0/es/

2.1.- Instalación y configuración.

Vamos a realizar la instalación de un servidor Apache en una máquina con la distribución Debian 6.0.1 Squeeze.

Empezamos por identificarnos en la máquina con el usuario root y, a continuación, ejecutamos:

```
# apt-get install apache2     (o apt install apache2)
```

Debido a que pretendemos montar una plataforma LAMP, por sus ventajas derivadas de las características del software libre, instalaremos también los siguientes componentes: MySQL y PHP.

```
# apt-get install php5 mysql-client mysql-admin mysql-query-browser phpmyadmin
```

Una vez instalado, para verificar si funciona, podemos hacerlo desde un navegador, escribiendo en la barra de direcciones :

```
http://localhost ó http://127.0.0.1
```

o bien, si accedemos desde otro equipo de la red a la dirección IP de esta máquina, deberíamos obtener una página con el mensaje “It Works!”, confirmando así su correcto funcionamiento.

Otro método de operar es descargar el código fuente de la aplicación desde la web del proyecto Apache; luego descomprimir, compilar e instalar; realizar el proceso empleando los siguientes comandos:

```
cd /usr/local/src
wget http://apache.rediris.es//httpd/httpd-2.2.19.tar.gz
tar xvzf httpd-2.2.19.tar.gz
cd /usr/local/src/httpd-2.2.19
./configure --prefix=/usr/local/apache --enable-module=most --enable-mods-shared=most
make
make install
```

Por defecto, Apache sirve las páginas web que están en la carpeta “**/var/www/**”; si nos situamos en esa carpeta, encontramos un archivo “**index.html**” que es el que contiene el “It Works!”. En esta carpeta podemos crear nuevas carpetas en donde ubicaremos nuevas páginas web que deseamos servir, todas ellas accesibles a través del puerto 80.

Si la única pretensión es servir una página web, podemos integrar su contendido aquí. En caso que se pretenda servir más páginas web, es más recomendable la utilización de los **Hosts Virtuales**; para ello accedemos a la carpeta ”**/etc/apache2/sites-enabled**”, donde hay un fichero llamado ”**000-default**”, que nos va a servir de ejemplo para la creación de hosts virtuales, los cuales van a permitir servir varias web desde una sola dirección IP utilizando para cada una un puerto distinto.

Apache se configura colocando directivas en archivos de configuración de texto plano. El archivo principal de configuración se llama **apache2.conf**. Además, se pueden añadir otros archivos de configuración mediante la directiva ”**Include**”, y se pueden usar comodines para incluir muchos archivos de configuración. Todas las directivas deben colocarse en alguno de esos archivos de configuración. Apache2 sólo reconocerá los cambios realizados en los archivos principales de configuración cuando se inicie o se reinicie.

Como ya hemos comentado, el archivo de configuración predeterminado de Apache2 es **/etc/apache2/apache2.conf**. Se puede editar este archivo para configurar el servidor Apache2, para configurar el número de puerto, la raíz de documentos, los módulos, los archivos de registros, los hosts virtuales, etc. Pasamos a ver alguna de las principales directivas:

- **ServerTokens**, para configurar la cantidad de información que Apache aporta sobre sí mismo.

- **ServerSignature**, para indicar datos sobre Apache en el pie de los mensajes de error.
- **Alias** permite direccionar a una carpeta que puede estar fuera del árbol de directorios especificado

en **DocumentRoot**.

- **userDir** permite redireccionar al directorio personal del usuario si se recibe una solicitud de tipo

~usuario.

Para modificar el servidor virtual predeterminado, editamos el archivo /etc/apache2/sitesavailable/ default. En el caso de querer configurar un nuevo servidor o sitio virtual, copiaríamos ese archivo dentro del mismo directorio con el nombre que se haya elegido, y editaríamos el nuevo archivo para configurar el nuevo sitio usando algunas de las directivas que se describen a continuación:

- **ServerName**, en el caso de no tener un dominio registrado emplearíamos localhost.
- **CustomLog** define el archivo .log donde se guardan los logs de acceso.
- **ServerAdmin** especifica la dirección de correo del administrador del servidor. El valor por omisión es **webmaster@localhost**.
- **Listen** especifica el puerto (y, opcionalmente, la dirección IP) por el que escuchará Apache2. La directiva se puede encontrar y cambiar en su propio archivo de configuración, **/etc/apache2/ports.conf**.
- **DocumentRoot** especifica dónde Apache debe buscar los archivos que forman el sitio. El valor predeterminado es **/var/www**.
- **RedirectMatchen** en **/etc/apache2/apache2.conf**, las peticiones se redirigirán a **/var/www/apache2-default**, que es donde reside el sitio predeterminado de Apache2. Cambiar este valor en el archivo de host virtual implica crear ese directorio si fuese necesario.

En vídeo práctico se explica cómo instalar y configurar un servidor web Apache en una máquina virtual con el sistema operativo Ubuntu con Webmin:
http://www.youtube.com/watch?feature=player_embedded&v=p1kDRqd_JHg

2.2.- Iniciar Apache.

Fuentes:

http://httpd.apache.org/docs/2.0/es/

https://openwebinars.net/academia/aprende/servidor-apache/

Comenzamos a trabajar con el servidor web instalado. Si hemos instalado Apache en la ruta **/usr/local/apache**, podemos probar su configuración por defecto e intentar iniciar el servicio de la siguiente forma:

```
/usr/local/apache/bin/apachectl configtest
```

Si todo está correcto debería devolver un mensaje del tipo ”Syntax Ok”

usr/local/apache/bin/apachectl start y el servidor debería estar arrancado, con lo cual, si en un navegador introducimos la URL: http://localhost veríamos la página de bienvenida de Apache.

Si el puerto especificado en la directiva Listen del fichero de configuración es el que viene por defecto, es decir, el puerto 80 (o cualquier otro puerto por debajo del 1024), entonces es necesario tener privilegios de usuario root (superusuario) para iniciar Apache, de modo que pueda establecerse una conexión a través de esos puertos privilegiados. Una vez que el servidor Apache se ha iniciado y ha completado algunas tareas preliminares, tales como abrir sus ficheros log, lanzará varios procesos, procesos hijo, que hacen el trabajo de escuchar y atender las peticiones de los clientes. El proceso principal, httpd, continúa ejecutándose como root, pero los procesos hijo se ejecutan con menores privilegios de usuario.

El demonio httpd se debería invocar empleando el script de control apachectl, que es el que se encarga de fijar variables de entorno y pasa al demonio (httpd) cualquier opción que se le pase cómo argumento por línea de comandos.

El script apachectl es capaz de interpretar los argumentos start, restart, y stop y traducirlos en las señales apropiadas para httpd.

Si en cualquier momento deseásemos parar, reiniciar o arrancar el servidor, podríamos emplear los siguientes comandos respectivamente:

```
# /etc/init.d/apache2 stop
# /etc/init.d/apache2 restart
# /etc/init.d/apache2 start
```

Una vez instalado el servidor Apache, es necesario acceder a su funcionalidad y gestionarlo como si de un servicio se tratase, de modo, que cuando establecemos cambios en su configuración, los mismos se vean reflejados.

¿Será necesario reiniciar el servicio Apache si, mediante la creación de un host virtual, hemos cambiado el puerto por el que escucha?

2.3.- Más sobre Apache.

En el próximo tema veremos cómo configurar Apache para disponer de varios servidores web virtuales y servidores web seguros así como otras utilidades.

También seguiremos el curso de OpenWeminars

3.- Aplicaciones web y servidores de aplicaciones.

Se define una aplicación web como una aplicación informática que se ejecuta en un entorno web, de forma que se trata de una aplicación cliente-servidor junto con un protocolo de comunicación previamente establecido:

- Cliente: navegador.
- Servidor: servidor web
- Comunicación: protocolo HTTP

Un servidor de aplicaciones es un software que proporciona aplicaciones a los equipos o dispositivos cliente, por lo general, a través de Internet y utilizando el protocolo HTTP. Los servidores de aplicación se distinguen de los servidores web en el uso extensivo del contenido dinámico y por su frecuente integración con bases de datos.

Un servidor de aplicaciones también es una máquina en una red de computadores que ejecuta determinadas aplicaciones, gestionando la mayor parte de las funciones de acceso a los datos de la aplicación.

Las principales ventajas de la tecnología de los servidores de aplicaciones es la **centralización y disminución de la complejidad en el desarrollo de las aplicaciones**, ya que no necesitan ser programadas, sino que son ensambladas desde bloques provistos por el servidor de aplicación.

Otra de las ventajas es la **integridad de datos y código** ya que, al estar centralizada en una o un pequeño número de máquinas servidoras, las actualizaciones están garantizadas para todos los usuarios.

El término servidor de aplicaciones se aplica a todas las plataformas. Dicho término se utiliza para referirse a los servidores de aplicaciones basadas en web, como el control de las plataformas de comercio electrónico integrado, sistemas de gestión de contenido de sitios web y asistentes o constructores de sitios de Internet.

Uno de los ejemplos destacados es el de Sun Microsystems, la plataforma J2EE. Los servidores de aplicaciones Java se basan en la Plataforma Java TM 2 Enterprise Edition (J2EE TM). J2EE utiliza un modelo de este tipo y en general, incluye un nivel Cliente, un nivel Medio, y un EIS. El servidor de tipo Cliente puede contener una o más aplicaciones o navegadores. La Plataforma J2EE es del Nivel Medio y consiste en un servidor web y un servidor EJB. (Estos servidores son también llamados “contenedores”.) También podría haber subniveles adicionales en el nivel intermedio. El nivel del SistemaEnterprise Information System (EIS, o “Sistema de Información Empresarial”) contiene las aplicaciones existentes, archivos y bases de datos.

Esta web detalla 120 aplicaciones disponibles gratuitamente vía web en donde se especifica la función de cada una de ellas.
http://especial.wetpaint.com/page/120+soluciones+gratis+web+2.0

Ampliaremos más en el tema 3 en el que estudiaremos Tomcat.