Despliegue de Aplicaciones Web (DAW).

Tema 1. Introducción a los servicios de red, Internet y Web

1. Tema 1.Introducción a los servicios de red, Internet y web.

1.1. Introducción

En la actualidad la manera en que los usuarios usan Internet está evolucionando muy rápidamente debido al desarrollo de nuevos servicios que están simplificando su uso. Esto posibilita el acceso a la información de una manera más eficiente.

Internet es una red que está enfocada al intercambio de información entre usuarios y equipos. Actualmente, la información disponible es enorme por lo que debe estar convenientemente organizada y estructurada para que sea accesible. La información se encuentra principalmente estructurada en páginas de hipertexto con enlaces a otras páginas, imágenes, vídeos, etc.

En general, las aplicaciones que hacen uso de Internet se basan en transferencia de información, distinguiéndose unas de otras por su formato o por los tipos de datos que manejan.

1.2. Estructuras de las redes de comunicaciones.

Una red de comunicación o de transmisión de datos es una estructura formada por medios *físicos* y *lógicos* (programas de transmisión y control) que permite la comunicación en una zona geográfica.

Las redes de comunicación permiten el intercambio de información entre un emisor y un receptor. A la señal recibida por el receptor le acompaña un componente de ruido que se suma durante su circulación a través de la red, por lo que habrá que introducir mecanismos de detección y corrección de errores.

Hay que tener en cuenta que una línea de comunicación no es solamente un cable, sino que también entran en juego otros elementos, como el *sistema de conmutación*, que decide la ruta que va a seguir la información hasta su destino y el *sistema de señalización*, que controla las comunicaciones establecidas.

1.3. Arquitectura TCP/IP.

Para interconectar sistemas, se necesitan una serie de programas que contienen todos los pasos que se deben seguir para establecer las conexiones, realizar transferencia de información, controlar los errores, etc. Este conjunto de normas resulta muy importante ya que sin ellas no estarán disponibles los servicios necesarios a bajo y alto nivel.

La mayoría de las redes, incluida Internet, usan la arquitectura TCP/IP para las comunicaciones. La especificación **TCP/IP** (*Transmission Control Protocol/Internet Protocol* o *Protocolo de Control de la Transmisión/Protocolo de Interred*) se encuentra definida en documentos oficiales denominados RFC (*Request for Comments* o *Petición de Comentarios*), definidos desde el año 1983 por el **IAB** (*Internet Activities Board* o *Comité de Arquitectura de Internet*).

TCP/IP proporciona una estructura y una serie de normas de funcionamiento para interconectar sistemas. Para simplificar esta tarea se realiza una subdivisión del trabajo en niveles o capas relacionadas entre sí de manera que cada capa realiza una labor determinada.

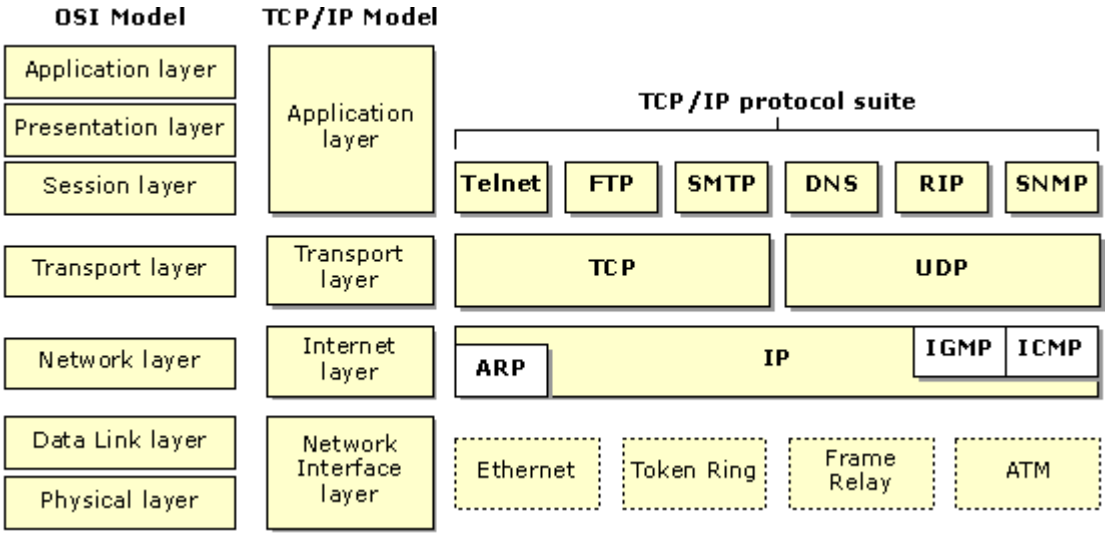
En cada capa existen una serie de protocolos que fijan unas series de normas para establecer la comunicación entre los sistemas. Cada protocolo se apoya en los protocolos de las capas inferiores para realizar su tarea y ofrece sus servicios a las capas superiores.

Las reglas que definen el protocolo TCP/IP son independientes del ordenador y del sistema operativo instalado, por lo que cualquier ordenador se puede comunicar con otro a través de la red. Ésta es una de las características que más ha contribuido al éxito y expansión de Internet.

En el caso de los sistemas operativos como Unix o Linux, TCP/IP está perfectamente integrada fundamentalmente por cuestiones históricas. (La red ARPANET se diseñó inicialmente para comunicar máquinas Unix.) Otros, como Windows, la adaptación se realizó de forma distinta. Microsoft desarrolló inicialmente una serie de protocolos y normas que definen una red de comunicación -la Red Microsoft- pero que era totalmente incompatible con Internet. Hoy en día, Windows incluye TCP/IP y muchos de los protocolos diseñados inicialmente para la Red Microsoft se han sustituido por otros para conseguir una integración completa.

Por otro lado el modelo OSI (***O**pen **S**ystem **I**nterconnection*) es el modelo de red descriptivo creado por la Organización Internacional para la Estandarización (ISO). Fue propuesto como una aproximación

teórica y como una primera fase en la evolución de las redes de ordenadores. Pero el modelo TCP/IP es el que realmente se usa.



Comparativa de arquitecturas OSI y TCP/IP

1.3.1. El modelo cliente-servidor.

La mayoría de los servicios ofrecidos por una red de comunicación de ordenadores se basan en el modo cliente-servidor, consistente en que un proceso cliente solicita un servicio y un proceso servidor presta el servicio al cliente que lo solicitó.

Otro modelo es el modelo entre pares o P2P (*Point to Point*), donde todos los equipos de la red son responsables por igual en la comunicación de las aplicaciones sin que exista un elemento que centralice las comunicaciones.

En el modelo cliente-servidor existen dos procesos que interactúan entre sí

- El proceso cliente, que normalmente inicia la comunicación, envía una petición a un proceso servidor quedando a la espera de una respuesta.
- El proceso servidor que suele ser un proceso que permanece a la espera escuchando las posibles conexiones de los clientes. Suelen ser procesos complejos, que necesitan de mecanismos de autenticación, autorización y seguridad; y robustos ya que habitualmente deben gestionar peticiones simultáneas de varios clientes.

1.4. Servicios de red.

Un servicio de red es una función que ofrecen las aplicaciones y los protocolos a los usuarios o a otras aplicaciones. Las aplicaciones se comunican e intercambian información con otras aplicaciones apoyándose en los protocolos TCP/IP, tanto en los protocolos de nivel de aplicación como en los de niveles inferiores.

Por ejemplo, en el servicio web, una aplicación instalada por el usuario (un navegador) se comunica con una aplicación en un servidor (Apache), las cuales se sirven del protocolo http para realizar la comunicación entre ambas.

1.4.1. Servicios de alto nivel.

Los servicios de alto nivel son aquéllos que manejan directamente los usuarios. Por ejemplo, en una red de telefonía, el servicio fundamental para los usuarios es la transmisión de voz. Análogamente, en una red de ordenadores sería el intercambio de archivos, mensajes, vídeos, etc. Los servicios de alto nivel más importantes que ofrece actualmente Internet son:

Transferencia de archivos entre equipos. Es uno de los servicios de transmisión de datos más usados en una red. El protocolo **FTP** (File Transfer Protocol o Protocolo de Transferencia de Ficheros) permite la transferencia de archivos entre ordenadores en redes de arquitectura TCP/IP.

Correo electrónico y mensajería instantánea. El servicio de **correo electrónico** (*e-mail*) consiste en el envío y recepción de mensajes de texto (además de un conjunto de *archivos adjuntos*) desde un usuario origen a otro destino, sin necesidad de que el destinatario se encuentre conectado y disponible para su recepción. Los servicios de **mensajería instantánea** se diferencian del correo electrónico en que los usuarios deben estar conectados y "en línea". Existen muchos sistemas de mensajería instantánea que funcionan actualmente en Internet, aunque en la actualidad los más usados son los ofrecidos a través de páginas web dinámicas.

Acceso remoto a equipos. El acceso vía **terminal remoto** a un ordenador ha sido el modo más frecuente de comunicaciones en red. Un programa *emulador de terminal* envía las órdenes que escribe el usuario en el *terminal* para que se ejecuten en el servidor, y este último devuelve los resultados para que aparezcan en la pantalla del usuario.

Uno de estos protocolos aunque actualmente menos utilizado es **Telnet** (*Telematics Network*). Es un protocolo simple de terminal remoto que permite establecer una conexión entre un usuario y un servidor. El usuario realiza pulsaciones sobre el teclado del terminal que son enviadas al servidor por la red y procesadas por éste, tras lo cual, se realiza el envío de nuevo al usuario del resultado de la ejecución de las órdenes que aparece sobre la pantalla del terminal. Por eso las órdenes que reconoce el protocolo *telnet* son las mismas que se manejan en el sistema operativo del servidor.

También existen otros protocolos para el acceso remoto a un equipo, como **SSH** (*Secure Shell* o *Interfaz Segura*), que funciona de la misma forma que *telnet* pero que también ofrece mecanismos avanzados de seguridad.

Otros protocolos de terminal remoto permiten también la conexión a través del entorno gráfico, dando al usuario la sensación de que está trabajando directamente con el equipo remoto.

Consulta de información en hipertexto. La **WWW** (*World Wide Web* o *Telaraña Mundial*) se utiliza para acceder a información distribuida a través de todos los servidores de Internet. Se accede a través de documentos llamados **páginas**. Cada página puede contener texto o imágenes gráficas, además de enlaces a otras páginas distintas; a este formato de documento se le llama **hipertexto.** Para poder ver correctamente estas páginas, se necesita un programa adecuado llamado **visor** o **navegador**.

Cuando un cliente escribe una dirección de una página web, el servidor correspondiente recibe una petición a través del protocolo TCP por el puerto 80 solicitando la página concreta. Este protocolo se llama **HTTP** (*HyperText Transfer Protocol* o *Protocolo de Transferencia de Hipertexto*).

Otros servicios, como la transferencia de archivos o el correo electrónico, se están adaptando a la WWW, de forma que es posible descargar un archivo, consultar el correo o publicar un mensaje en un foro desde una página de hipertexto. Esto ha sido posible gracias al desarrollo de lenguajes más potentes que permiten dinamizar la visualización de las páginas.

Los visores HTML actuales son capaces de interpretar código escrito en otras versiones del lenguaje que extienden la funcionalidad del protocolo, como XML, JavaScript, applets escritos en Java o PHP.

1.4.2. Servicios de bajo nivel.

Para que una red pueda ofrecer una serie de servicios de alto nivel, necesita de una compleja infraestructura que incluye a una serie de programas y servicios que realizan tareas más simples.

Cuando un usuario descargar una página de hipertexto, utiliza el servicio correspondiente de alto nivel, que estará disponible a través de alguna aplicación instalada en el equipo (como por ejemplo un navegador). Ésta requiere, a su vez, de la realización de otras operaciones más sencillas que tienen que ver con la forma en la que la red de comunicación subyacente transfiere los datos; como por ejemplo la comprobación de que el otro equipo está listo, la selección de la mejor ruta que debe seguir la información hasta alcanzar el destino, la confirmación de que el otro equipo acepta la conexión, la división de la información en fragmentos más pequeños para ser enviados individualmente, la ordenación y fusión de los mensajes recibidos en el destino, la comprobación de errores, etc. Estas operaciones se realizan gracias a la existencia de **servicios de bajo nivel**.

Los servicios de bajo nivel dependen de la red de comunicación sobre la que funcionan. Por ejemplo, para copiar un archivo en una red local, los servicios de bajo nivel deben obtener la dirección del equipo destinatario, pero no necesitan seleccionar la mejor ruta (porque sólo existe una). Sin embargo, para enviar un archivo a través de una conexión telefónica, el equipo debe realizar primero una llamada de teléfono al nº del proveedor de acceso, establecer la conexión y seleccionar la mejor ruta hasta el destino.

En Internet se utilizan dos servicios muy importantes que dan apoyo a otros servicios de alto nivel: **DHCP** (*Dynamic Host Configuration Protocol* o Protocolo de Configuración Dinámica de Equipos) y **DNS** (*Domain Name System* o Sistema de Nombres de Dominio). DHCP se usa para facilitar la configuración automática de los equipos, de forma que los usuarios poco experimentados o pardillos no tengan que configurar manualmente sus equipos. DNS se utiliza para ocultar el esquema de direcciones IP que usa Internet y así los usuarios puedan trabajar de una forma más cómoda con nombres de equipos.

1.5. El protocolo IP.

Las funciones principales del protocolo IP son establecer una dirección para cada nodo de la red y determinar las rutas o encaminamientos que se ejecutan en los dispositivos que interconectan las redes.

La comunicación a nivel IP se realiza mediante unidades de datos denominadas *datagramas* las cuales tienen un formato que está especificado en el protocolo IP.

IP dispone de dos versiones: la versión 4 (IPv4), la más utilizada actualmente, y la versión 6 (IPv6) que aunque está desarrollada hace tiempo, es ahora cuando empieza realmente a implantarse.

1.5.1. Direccionamiento IP.

IP permite la conectividad extremo a extremo en la comunicación. Esto implica que todos los dispositivos deben tener una dirección única en la red. El direccionamiento es independiente del dispositivo físico asignado y se puede modificar cuando se necesite.

El direccionamiento se establece mediante la *dirección IP*. Una dirección IP no identifica a un ordenador en la red, sino que identifica a una *interfaz de red*. Un ordenador con varias interfaces puede por tanto conectarse a diferentes redes. Incluso una misma interfaz de red puede tener varias direcciones IP; es lo que se conoce como direccionamiento virtual.

En IPv4, una dirección IP es un número binario de 32 bits, lo que permite un espacio de direcciones de 2^{32}; es decir (4.294.967.296) direcciones diferentes. Para un manejo más cómodo, las direcciones se dividen en 4 grupos de 8 bits, donde cada grupo se escribe en decimal y separados por un punto. Por ejemplo 80.36.234.12.

Para conseguir que las direcciones IP expresen información de direccionamiento y de encaminamiento se desglosan en dos partes:

- Identificador de red, que determina la red en que se encuentra el dispositivo.
- Identificador de equipo o *host* dentro de la red.

De esta manera, todos los equipos de una misma red comparten la parte de identificación de red, siendo éste un valor que dependerá del tamaño de la red. Las redes grandes usarán un identificador de red pequeño, y las redes pequeñas usarán uno grande.

La *máscara de red* permite diferenciar el prefijo de la dirección IP que se corresponde con el identificador de red, de la parte correspondiente al identificador de equipo o *host*. La máscara de red es un número de 32 bits en donde las posiciones a "1" definen la parte correspondiente a la identificación de red, y las posiciones a "0" la parte que corresponde al identificador de equipo.

	11001100 . 00110011 . 10101010 . 01010101	Dirección IP
AND	11111111 . 11111111 . 11111111 . 00000000	Máscara de red
	11001100 . 00110011 . 10101010 . 00000000	
	Identificador de Red	Identificador de Equipo

En la tabla siguiente se resumen las máscaras de red utilizadas para las divisiones más comunes.

Dirección y máscara	Número de red	Número de equipo
80.36.234.12 255.0.0.0	80	36.234.12
10.245.132.65 255.255.0.0	10.245	132.65
122.76.211.89 255.255.255.0	122.76.211	89

La máscara de red también puede expresarse mediante la notación CIDR (*Classless Inter-Domain Routing)* situando a continuación de la IP un sufijo que indica cuántos bits de la máscara de red están a "1". El último ejemplo de la tabla anterior se podría expresar de la forma:

122.76.211.89/24

De todo el conjunto de direcciones IP, algunas tienen un significado especial:

- **Dirección de red**. Identifica a toda una red. La parte correspondiente al identificador de host tiene todos sus bits a cero. Por ejemplo, la IP 122.76.211.89/24 tiene la dirección de red 122.76.211.**0**/24
- **Dirección de difusión limitada**. Se usa para mandar un mensaje de difusión o *broadcast* a todos los dispositivos de la propia red. Para todas las redes es la misma: 255.255.255.255
- **Dirección de difusión dirigida**. Se usa para mandar un mensaje de difusión o *broadcast* a todos los dispositivos de una red concreta. Esta dirección se compone por el identificador de red en la que se va a realizar la difusión, seguida del identificador de equipo con todos sus bits a "1". Por ejemplo la dirección 122.76.211.**255**/24 permite realizar difusión dirigida en la red 122.76.211.**0**/24
- **Dirección de bucle local**. Permite referenciar de forma interna al interfaz. Esta dirección se usa para los procesos TCP/IP que se generan dentro del equipo. Se usa cualquier dirección de la red 127.0.0.0/8, por ejemplo la dirección 127.0.0.1/8.

Dentro del espacio de direcciones, algunas están reservadas para el ámbito privado. Estas *direcciones privadas* que no deben tener acceso a Internet se usan para redes privadas o *intranets* y deben pertenecer a las siguientes redes: 10.0.0.0/8, 172.16.0.0/16 y 192.168.0.0/16. En contra posición, las llamadas *direcciones públicas* permiten identificar a un dispositivo conectado a Internet.

También existe un conjunto de direcciones reservado para aquellas redes en donde no existe ningún tipo de direccionamiento definido. En este caso, son los propios equipos los que se asignan a sí mismo las direcciones usando las direcciones de la red 169.254.0.0/16.

A parte de la configuración IPv4 vista anteriormente es posible que se necesite una configuración IPv6. IPv6 admite un rango de direcciones mucho mayor, pero tiene un formato más complejo. Utiliza un formato de 128 bits, permitiéndose alrededor de 340 sextillones de direcciones.

La forma de especificar direcciones IPv6 es x:x:x:x:x:x:x:x donde cada x representa un número de 16 bits normalmente expresado con 4 dígitos hexadecimales. Por ejemplo:

2001:0db8:85a3:08d3:1319:8a2e:0370:7334

Las direcciones IPv6 también, se pueden representar de otras formas diferentes:

- Igual que la anterior, pero suprimiendo todos los ceros consecutivos en la dirección y sustituyéndolos por "::". Estos "::" solamente pueden aparecer una vez en la dirección.
- Utilizando una notación mixta formada por una parte de dirección v6 (seis números hexadecimales de 16 bits separados por dos puntos) y otra de v4 (cuatro números decimales de ocho bits).

Una dirección IPv6 también tiene varios campos: los primeros bits forman un **prefijo**, que indica el tipo de dirección; los bits centrales especifican un número de red (que puede no existir) y los bits finales especifican un número de estación.

Prefijo	Descripción
00	Dirección IPv4.
2 ó 3	Dirección asignada por un proveedor de acceso a Internet.
de FE80 a FEBF	Direcciones privadas dentro del ámbito de la subred.
de FEC0 a FEFF	Direcciones privadas dentro del ámbito de la red (intranet).
FF	Dirección de multidifusión.

En IPv6 la representación de las máscaras de red es justamente la contraria a IPv4: un sufijo /50, por ejemplo, indica que los 50 últimos bits de la dirección se reservan para numerar estaciones.

Para mantener la coexistencia entre IPv4 e IPv6, los equipos tienen instalados los dos protocolos a la vez, asignándose una dirección IPv4 y otra IPv6. Este mecanismo permite la coexistencia hasta que IPv6 se haya implantado definitivamente, pero aumenta la complejidad en la configuración de red de los equipos.

1.5.2. Encaminamiento IP.

El protocolo IP también es el responsable del encaminamiento de los datagramas, llevándolos desde una máquina origen a otra destino independientemente de la red en que se encuentren las máquinas.

Los *encaminadotes* o *routers* son los dispositivos encargados de esta tarea, manteniéndose conectados al menos a dos redes y realizando el encaminamiento de los datagramas que pasen por él. Para ello se apoyará en las *tablas de encaminamiento*.

Las tablas de encaminamiento almacenan la información necesaria para realizar el encaminamiento de los datagramas y están presentes tanto en los routers como en los equipos.

Los propios equipos también realizan tareas de encaminamiento en su tráfico saliente, de forma que cuando un equipo coloca un datagrama en la red ya ha decidido si el datagrama lo envía directamente al equipo destino porque está en la misma red, o lo envía al encaminador para que lo encamine hacia su destino si no está en la misma red.

Cuando un router recibe un datagrama, si éste se dirige a una dirección que pertenece a una red a la que está conectado, realiza una entrega directa; en el caso de que vaya dirigida a una red a la que no está conectado directamente, lo reenviará al router que indique su tabla de encaminamiento. Este proceso se repetirá hasta que se alcance la red de destino o agotar el tiempo de vida del datagrama.

Cuando se arrancan los equipos, las tablas de encaminamiento de los routers y de los equipos se inicializan con las rutas correspondientes a sus redes adyacentes. A partir de este momento, la configuración de las tablas de encaminamiento se puede realizar de dos formas: encaminamiento *estático* o *dinámico*.

- En el encaminamiento estático las tablas se configuran manualmente. No es muy recomendable esta estrategia ya que cualquier cambio en la estructura de la red necesita de una actualización de las tablas.
- En el encaminamiento dinámico es el propio encaminador el que actualiza sus tablas usando para ello protocolos que permiten que los routers intercambien información entre ellos para mantener actualizadas sus tablas. Protocolos de encaminamiento son RIP (*Routing Information Protocol*), OSPF (*Open Shortest Path First*) y BGP (*Border Gateway Protocol*).

Los equipos de aquellas redes que no se conectan con otras, salvo a Internet, también necesitan conocer a dónde deben enviar los datagramas que no tienen como destino la propia red. A este parámetro se le denomina *puerta de enlace, pasarela por defecto* o *default gateway*. Su valor se establece con la dirección IP del dispositivo que conecta nuestro equipo con Internet (normalmente es la dirección del encaminador o el dispositivo propiedad del proveedor de acceso que conecta la red de comunicación de nuestro equipo con Internet).

1.6. Protocolos TCP y UDP.

IP permite comunicar dos máquinas haciendo que los datagramas puedan ir desde un origen a un destino, pero no permite mantener varias comunicaciones simultáneas entre dichos equipos, ya que sólo con la IP no se puede diferenciar qué datagramas pertenecen a cada una de las comunicaciones establecidas.

La capa de transporte permite diferenciar y gestionar múltiples orígenes y destinos en una comunicación, y múltiples comunicaciones en cada equipo. Para ello se usa el concepto de *puerto de comunicaciones* para identificar los procesos de nivel de aplicación entre los que está establecida la comunicación.

Todo proceso de la capa de aplicación usa uno o varios puertos a través de los cuales es accesible. Cada puerto se identifica por un número binario de 16 bits cuyos valores en decimal varían entre o 65535.

Según los servicios que usan los puertos, se pueden diferenciar en tres clases:

- [0-1023]. Son los llamados puertos conocidos *(well known ports*) y están reservados para aplicaciones y servicios estándar: FTP, SMTP, HTTP, etc. Las aplicaciones clientes se conectan a estos puertos para acceder a los servicios.
- [1024-49151]. Puertos registrados usados para aplicaciones no estándar instaladas por el usuario que no tienen un puerto bien conocido asignado.
- [49152-65535]. Puertos dinámicos. Se emplean para iniciar conexiones desde el cliente y no suelen usarse en procesos de servidor.

La correspondencia entre los procesos y los puertos a usar se pueden establecer de dos maneras:

- Asignación estática: los puertos conocidos se reservan para aplicaciones estándar y sólo puede emplear por estos procesos.
- Asignación dinámica: cuando un proceso necesita un puerto, y éste no se asigna de forma estática, el sistema operativo le asigna uno que esté disponible.

Aunque a nivel de transporte existen los protocolos TCP y UDP, ambos son independientes y los puertos que usan también lo son.

Es posible controlar la información que circula a través de los puertos utilizando un *cortafuegos* o *firewall*, que se instala en el propio equipo o que funciona de forma autónoma entre el equipo y la red. También existen dispositivos encaminadores o *routers* que pueden realizar funciones de cortafuegos.

Cuando se instala un servicio en un equipo siempre es necesario que el puerto o puertos asociados se encuentren abiertos o activos. Por defecto, el equipo activa los puertos cuando se pone en marcha el servicio, de forma que el resto de ordenadores de la red pueden utilizarlo.

La tabla muestra una lista con algunos de los puertos de transporte más utilizados

Puerto	Servicio	Descripción
7	Echo	Puerto de "eco"
20	FTP Datos	Protocolo FTP de transferencia de archivos
21	FTP Control	Protocolo FTP de transferencia de archivos
22	SSH	Protocolo de terminal remoto seguro
23	Telnet	Protocolo de terminal remoto
25	SMTP	Protocolo de transferencia de correo
42	Nameserver	Servidor de nombres
43	WHOIS	Protocolo de información
53	DNS	Servidor de nombres
67	Bootp Servidor	Protocolo de arranque a través de la red
68	Bootp Cliente	Protocolo de arranque a través de la red
69	TFTP Protocolo	FTP trivial
79	Finger	Información sobre usuarios en equipos
80	HTTP	Protocolo de transmisión de hipertexto
88	Kerberos	Protocolo de autentificación seguro
109	POP2	Protocolo de gestión de correo electrónico
110	POP3	Protocolo de gestión de correo electrónico
137-139	NetBIOS	Protocolo de la red Microsoft
143	IMAP	Protocolo de correo electrónico
161, 162	SNMP	Protocolo de administración de red
194	IRC	Internet Relay Chat
256	SNMP	Protocolo de administración de red
389	LDAP	Protocolo de localización de servicios
443	HTTPS	Protocolo HTTP seguro (cifrado)
445	SMB CIFS	Protocolo de la red Microsoft
513	Rlogin	Protocolo de terminal remoto
514	Rshell	Protocolo de terminal remoto
515	Impresora TCP/IP	Protocolo de una impresora de red
520	RIP	Protocolo de encaminamiento
524	NCP	Protocolo de la red Novell NetWare
1080	Sockets	Servidor proxy
989	FTP Datos sobre SSL	Protocolo FTP seguro (cifrado)
990	FTP Controlsobre SSL	Protocolo FTP seguro (cifrado)
992	Telnet sobre SSL	Protocolo Telnet seguro (cifrado)
993	IMAP sobre SSL	Protocolo IMAP seguro (cifrado)
1745	Winsock-proxy	Servidor proxy de Microsoft
3306	MySQL	Base de datos relacional
2049-4045	NFS	Sistema de archivos de red de Linux
6000-6007	X-Window	Interfaz gráfica de Unix/Linux

Para poder conocer el estado de actividad de los puertos de comunicación se utilizan diferentes herramientas que envían mensajes a un rango de puertos especificado y comprueban si éstos están activos. Software de este tipo son *nmap* para sistemas Linux, o *GFI LANguard* para Windows.

Por regla general, cuantos más puertos de comunicación se encuentren activos o abiertos en nuestro equipo, mayor será el riesgo a padecer un problema de seguridad.

1.6.1. Protocolo UDP.

UDP (*User Datagram Protocol*) es un protocolo no orientado a la conexión. Esto implica que no se realiza una conexión previa a la transmisión y que no existe un control de flujo de forma que se pueden entregar datagramas duplicados o desordenados. Hay que tener en cuenta que el receptor solo conoce la IP del emisor.

Se suele usar cuando es más importante la velocidad de la transmisión que la fiabilidad de la misma (streaming, voz IP) o en protocolos de tipo petición respuesta, como son DHCP y DNS.

1.6.2. Protocolo TCP.

TCP (*Transmisión Control Protocol*) proporciona un servicio orientado a la conexión, lo que implica establecer una conexión antes de comenzar la transmisión y llevar un control de flujo y de errores. Con ello se consigue una transmisión fiable, aunque más lenta que con UDP.

Cuando se desea conectar con un servicio de un equipo, hay que iniciar una conexión TCP. Una vez establecida, cualquiera de los extremos puede empezar a transmitir y también terminarla cuando lo desee. La conexión se define de forma única indicando la dirección IP y número de puerto del equipo origen, y la dirección IP y número de puerto del equipo destino. A este par formado por dirección IP y número de puerto se le llama *conector* o *socket*. Se dice que un puerto de comunicación está *abierto* o *a la escucha* cuando existe un programa en el equipo que controla las comunicaciones que llegan a través del puerto.

1.7. Los Sistemas Operativos.

Un **sistema operativo** es un conjunto de programas que funcionan sobre una computadora y que se encarga de administrar los recursos de ella. Además, el sistema operativo también incluye un conjunto de rutinas básicas que facilitan la tarea de desarrollo de aplicaciones.

Desde el punto de vista de la gestión sobre los recursos de red, podemos dividir a éstos en dos tipos: **sistemas operativos de cliente** y **sistemas operativos de servidor**.

Un *sistema operativo de cliente* es aquél que no ha sido desarrollado con soporte para la administración de los recursos de la red aunque sí puede acceder a ellos como recursos de trabajo. Habitualmente requieren un menor espacio de almacenamiento en disco y menores requerimientos de memoria y capacidad de proceso. Por su parte, un *sistema operativo de servidor* es aquél especialmente diseñado para trabajar en una red y permitir la administración eficiente de sus recursos.

Aunque esta división todavía es aplicable a los sistemas operativos, resulta cada vez más confusa, ya que los sistemas operativos de cliente y servidor llegan a diferenciarse solamente por las aplicaciones y servicios que tienen instalados, no por el núcleo del sistema.

1.7.1. Modelo cliente-servidor.

Cuando un equipo quiere acceder a los servicios disponibles en un servidor remoto, primero tiene que enviar un mensaje de solicitud y dirigirlo al puerto asociado a ese servicio. El servidor deberá tener activo ese puerto para recibir la solicitud, procesarla y enviar los resultados. En este modelo, el cliente debe conocer cuál es el número de puerto que tiene asociado ese servicio en el servidor, normalmente porque se trata de un *puerto bien conocido*.

El concepto de modelo cliente-servidor se puede aplicar tanto a programas que se ejecutan en un mismo equipo como a equipos conectados a través de la red. Este modelo no restringe la función que desempeña cada equipo en la red, de forma que un equipo se puede comportar como cliente de unos determinados servicios a la vez que comportarse como servidor de otros.

Este modelo funciona muy bien en redes de ordenadores donde los servicios se gestionan a través de servidores centralizados y donde pueden existir varios servidores que repartan el trabajo de diferentes tareas.

Las características que definen a un equipo como cliente son:

- Requiere de una potencia de cálculo menor, aunque esto dependerá de los programas que ejecute.
- Es utilizado por los usuarios para realizar su trabajo diario.
- Es el encargado de iniciar las peticiones o solicitudes.
- Recibe las respuestas de los servidores, obteniendo la información que ha solicitado o un mensaje de rechazo.
- Puede realizar peticiones de varios servicios a diferentes servidores.

Las características que definen a un equipo como servidor son:

- Suele requerir una gran potencia de cálculo y memoria principal para poder atender todas las peticiones que recibe.
- Permanece a la espera de recibir peticiones. Cuando recibe una petición, la procesa y envía los resultados o un mensaje de rechazo.
- Suele aceptar un gran número de peticiones, aunque este valor puede limitarse.
- Es un equipo dedicado a atender peticiones y los usuarios no suelen trabajar con él directamente.

Entre las ventajas del uso del modelo cliente-servidor en los servicios de red podemos destacar:

- Se establece un mayor control de la seguridad y el acceso a servicios autorizados, ya que éste se realiza a través de cada servidor.
- Puede aumentarse fácilmente la capacidad de los equipos o su número.
- Permite un mantenimiento más sencillo y una división de responsabilidades entre los administradores, ya que los cambios solamente se deben realizar en los servidores involucrados.

El modelo cliente-servidor también tiene sus inconvenientes:

- Sobrecarga de los servidores cuando existen muchas peticiones.
- El mal funcionamiento de un servidor hace que no estén disponibles los servicios que ofrece.
- Los servidores requieren de sistemas operativos y programas muy estables.

1.7.2. Herramientas de administración de servicios.

Los sistemas operativos disponen de una serie de programas y herramientas que facilitan la instalación, configuración y administración de los servicios de que disponen.

1.7.2.1 Herramientas en Microsoft Windows

En los sistemas operativos Microsoft Windows 200X se utilizan las denominadas *Herramientas administrativas*. Algunas de las opciones que aparecen por defecto en el menú de *Herramientas administrativas* no están disponibles por defecto y se pueden añadir cuando se instalan los servicios correspondientes.

Algunas de las herramientas más importantes que incluye Windows 200X Server son las siguientes:

Administración de equipos: se usa para la administración del equipo local o equipos remotos. Desde aquí se puede realizar la gestión completa de un equipo e incluye las herramientas del sistema (*visor de sucesos*, *administrador de dispositivos*, etc.), almacenamiento (*desfragmentador de discos*, *administrador de discos*, etc.) y servicios y aplicaciones instaladas (*DHCP*, *IIS*, etc.).

Administrador de Internet Information Services (IIS): se usa para administrar el Servidor de Información de Internet (IIS).

Administrador del servidor: mediante Administre su servidor o el Asistente para configurar su servidor se puede instalar, desinstalar y configurar servicios en el equipo.

DHCP: se usa para configurar el servidor DHCP del equipo local.

DNS: se usa para configurar el servidor DNS del equipo local.

Directiva de seguridad local: se utiliza para establecer características de seguridad del equipo cuando éste no pertenece a ningún dominio.

Directiva de seguridad de dominio: se utiliza para establecer características de seguridad sobre el dominio.

Directiva de seguridad del controlador de dominio: se utiliza para establecer características de seguridad sobre el servidor del dominio.

Servicios: muestra una lista con todos los servicios iniciados en el sistema y los que están inactivos.

Terminal Services: incluye las herramientas para el acceso remoto a equipos.

Programador de tareas: se utiliza para definir qué programas se van a ejecutar en el equipo en una fecha o con una frecuencia determinada.

Visor de eventos o **sucesos**: muestra información sobre los sucesos, avisos o errores que se han producido mientras el equipo ha estado en funcionamiento.

1.7.2.2 Herramientas en GNU/Linux

Tradicionalmente, estas herramientas no han existido y todas las tareas se realizaban modificando los archivos de configuración correspondientes o usando órdenes. Esta técnica tiene la ventaja de que permite modificar cualquier parámetro, está más estandarizada y se puede utilizar de la misma forma en diferentes distribuciones y versiones (normalmente los nombres y las ubicaciones de los archivos pueden variar).

Actualmente, han aparecido una serie de herramientas que facilitan enormemente la configuración y administración del sistema. Algunas de ellas son muy dependientes de la distribución de Linux utilizada y puede que no incluyan todas las opciones de configuración.

Todas las modificaciones en la configuración del sistema deben realizarse utilizando la cuenta de usuario que tiene privilegios para realizar estas operaciones. Esta cuenta, que normalmente es **root**, dispone de acceso a las órdenes, archivos de configuración y herramientas gráficas de administración. Por seguridad, el uso de esta cuenta debe estar limitado a realizar este tipo de acciones y utilizar una cuenta de usuario normal para el resto de operaciones que no requieran de permisos especiales.

Si un usuario normal intenta acceder a cualquier herramienta que requiera privilegios de *root*, el sistema nos solicitará la contraseña de esta cuenta de administrador antes de acceder a ella. En caso de que necesitemos ejecutar una orden en la línea de órdenes con los privilegios de administrador, podemos cambiarnos a *root* utilizando la orden *su*.

A veces puede resultar muy útil la posibilidad de ejecutar determinados programas con los privilegios de *root*. Para ello, se pueden utilizar las órdenes *su*, *sudo* (que funciona con programas de la línea de órdenes), *kdesu* (que funciona con programas del entorno gráfico KDE) o *gksudo* (que funciona en el entorno Gnome). Si utilizamos muy a menudo un programa con los privilegios de *root*, podemos crear un acceso directo en el escritorio y así nos ahorraremos tener que escribir las órdenes necesarias.

Cada distribución Linux suele incorporar unas herramientas específicas de configuración. Pero una buena herramienta de configuración gráfica es **Webmin** que funciona en más de 35 sistemas Unix/Linux diferentes y es multilenguaje. La forma de acceder a Webmin una vez instalada, es a través de una ventana del navegador de Internet con dirección https://localhost:10000. Este método permite también acceder a Webmin desde otro equipo distinto al que se encuentra instalado.

1.7.3. Instalación de programas.

Cuando se necesita que un equipo ofrezca algún servicio, es necesario instalar en él todos los programas necesarios, que suelen estar disponibles a través de uno o varios archivos. Estos archivos suelen contener un programa de instalación que guía paso a paso, una herramienta de configuración del servicio, el proceso demonio encargado de atender las peticiones y varios archivos de configuración o valores del registro del sistema.

A la hora de instalar un determinado servicio en el sistema, podemos hacerlo de varias formas: abriendo el programa ejecutable de instalación y mediante la herramienta de instalación de programas.

1.7.3.1. Instalación de componentes en Microsoft Windows.

En Windows existen dos tipos de programas que pueden instalarse: los *componentes de Windows* (programas y utilidades que se incluyen en el disco de instalación del sistema operativo) y las *aplicaciones*.

Los componentes de Windows se instalan o desinstalan desde varias ventanas del sistema dependiendo de su tipo. Los componentes de red de Windows (servicios, protocolos y clientes) se gestionan desde las propiedades del icono “Mis sitios de red” del *Escritorio* o el *Menú de Inicio* (Windows 2000/XP/2003) o desde el *Centro de redes y recursos compartidos* (Windows Vista/7/2008).

El resto de componentes se instalan, al igual que las aplicaciones, desde el icono "Agregar o quitar programas" (Windows 2000/XP/2003) o "Programas y características" (Windows Vista/7/2008), accesibles desde el *Panel de control*.

1.7.3.2 Instalación de paquetes en GNU/Linux.

A la hora de instalar o desinstalar programas y aplicaciones en Linux, hay que conocer una serie de términos:

- Paquete: está formado por un conjunto de archivos o ficheros que se distribuyen conjuntamente, de forma que es la unidad mínima de instalación en un sistema operativo. Para evitar complejidad y facilitar las tareas de administración, no se permite que un paquete se instale parcialmente en el sistema. Es posible que un programa esté distribuido en varios paquetes y que un paquete contenga varios programas. El paquete se almacena en un archivo con la extensión concreta para identificar su tipo (por ejemplo, ".rpm" o ".deb") y contiene, además de los archivos a copiar, las rutas donde se copian, información sobre la persona que los ha desarrollado, una clave cifrada que asegura su autenticidad, la suma de verificación para comprobar si ha sido alterado, las operaciones para configurar el programa, etc.
- Repositorio o fuente de instalación: se trata de un gran almacén que guarda los archivos con los paquetes para instalar. La fuente de instalación más común siempre ha sido el soporte óptico en CD o DVD, pero actualmente han proliferado los repositorios oficiales y no oficiales disponibles en servidores de Internet.
- Dependencia: en muchas ocasiones, un paquete puede requerir de otro para poder funcionar correctamente (porque dispone de alguna librería o archivo que necesita). En este caso, es necesario que el paquete requerido sea instalado antes para poder mantener la estabilidad del programa que queremos instalar e incluso la del sistema.
- Conflicto: se producen cuando queremos instalar un paquete que no puede coexistir al mismo tiempo con otros paquetes a instalar o ya instalados. Ocurren normalmente porque se trata de programas que realizan funciones similares, con lo que su uso simultáneo puede acarrear problemas.
- Gestor de paquetes: es el programa encargado de gestionar los paquetes de instalación en el sistema, ofreciendo operaciones como: instalar, desinstalar y actualizar paquetes. Sus tareas son: gestionar una base de datos con información sobre los paquetes instalados y los disponibles para instalar. Selección de las fuentes de instalación. Consulta de información sobre los paquetes instalados y no instalados. Gestionar correctamente las dependencias.

Existen básicamente dos métodos para instalar programas en Linux:

A través de los paquetes fuente: consiste en utilizar paquetes que contienen los archivos con el código fuente del programa. Por lo tanto, en la operación de instalación del paquete es necesario copiar los archivos, compilarlos a código ejecutable y configurarlos convenientemente.

A través de paquetes precompilados: se utilizan paquetes que previamente han sido compilados en código ejecutable, lo que facilita enormemente las tareas de instalación automática. Sin embargo, es necesario disponer del paquete específico que ha sido diseñado para funcionar en una distribución concreta sobre un equipo con una arquitectura concreta.

Algunos de los gestores de paquetes más utilizados actualmente son *dpkg* (utilizado por Debian y otras distribuciones derivadas), *RPM* (es el sistema más extendido que se ha convertido en un estándar para las distribuciones Red Hat, Fedora, SuSE, Mandriva, etc.), *tgz* (usado por Slackware) y *Pacman* (usado en la distribución Arch).

Una herramienta que se puede utilizar para la gestión de paquetes es *Webmin*, que tiene la ventaja de que funciona en cualquier versión de Linux.

El gestor de paquetes *dpkg* se basa en el uso de los paquetes almacenados en archivos con extensión ".deb", que contienen los archivos que forman parte del programa, información de su desarrollador, etc.

Los paquetes ".deb" se gestionan a través de dos herramientas que funcionan en niveles diferentes: por un lado está **dpkg**, que funciona en un nivel más bajo, mientras que por otro lado está **apt**, que funciona en un nivel más alto. Esta jerarquía hace que *dpkg* proporcione todo lo necesario para manipular paquetes, mientras que *apt* utiliza la anterior para permitir que el usuario trabaje de una forma más cómoda, ofreciendo funciones para obtener paquetes desde diferentes lugares o resolver dependencias complejas.

El proyecto Debian mantiene, para cada una de las versiones que publica, tres tipos de distribuciones, que se utilizan en las fases de desarrollo: **Estable** (*stable*): es la distribución que se recomienda, ya que se han depurado todos los errores (o por lo menos todos los conocidos). **De prueba** (*testing*): es utilizada por los que quieren disponer cuanto antes de la última versión, aunque todavía no se han corregido algunos pequeños errores detectados. **Inestable** (*unstable*): es la distribución utilizada por los programadores que forman parte del proyecto Debian durante todo el desarrollo activo del sistema.

La herramienta **apt** conforma un entorno completo de gestión de paquetes que realiza de forma automática la mayoría de operaciones complejas para el usuario. Esta herramienta dispone de muchas órdenes que se pueden ejecutar en el intérprete:

Orden	Descripción
apt-cache	Manipula los archivos de los paquetes disponibles en cache
apt-cdrom	Permite incluir la unidad de CD-ROM como fuente de instalación de paquetes.
apt-config	Permite acceder a la configuración de las herramientas apt (archivo apt.conf).
apt-file	Muestra a qué paquetes pertenecen los archivos indicados.
apt-get	Permite la instalación, desinstalación y actualización de paquetes.
apt-key	Gestiona la lista de claves utilizada para autentificar los paquetes descargados para instalar.
apt-rpm	Se trata de una versión modificada de la herramienta apt que pretende funcionar con el gestor de paquetes RPM. (En desarrollo).
auto-apt	Instala paquetes de forma automática cuando se ejecuta un programa que necesita un determinado paquete.
localepurge	Elimina de forma automática aquellos archivos de la documentación del sistema que pertenecen a idiomas no utilizados por los usuarios.

Ejemplos de utilización de la herramienta apt:

- Para actualizar la base de datos de paquetes disponibles para instalación, nuevas versiones y actualizaciones de paquetes ya instalados, se usa esta orden con privilegios de administrador:

  ```
  # apt-get update
  ```

- Para actualizar con nuevas versiones los paquetes ya instalados

  ```
  # apt-get upgrade
  ```

- Para instalar el paquete "manolete" (si existieran dependencias, se resolverían confirmando el mensaje de advertencia que aparece):

  ```
  # apt-get install manolete
  ```

- Para reinstalar el paquete "manolete" dañado o actualizarlo con una nueva versión disponible:

  ```
  # apt-get  --reinstall install manolete
  ```

- Para instalar el paquete "manolete" en la última versión todavía de prueba:

  ```
  # apt-get install manolete /testing
  ```

- Para eliminar el paquete "manolete" y mantener los archivos de configuración. (*apt* se encargará de eliminar también los paquetes dependientes, después de recibir confirmación):

  ```
  # apt-get remove manolete
  ```

- Para eliminar el paquete "manolete" y todos los archivos de configuración:

  ```
  # apt-get purge manolete
  ```

- Para actualizar a una nueva versión de Linux disponible (debe estar disponible la fuente donde se encuentran los paquetes de la nueva versión):

  ```
  # apt-get –u dist-upgrade
  ```

- Para eliminar los paquetes temporales que han sido descargados al equipo:

  ```
  # apt-get clean
  ```

- Para eliminar los paquetes antiguos que han sido descargados al equipo:

  ```
  # apt-get autoclean
  ```

- Para solicitar a *apt* que realice una recomendación para instalar, desinstalar y eliminar paquetes, de forma que se tengan en cuenta las sugerencias incluidas con los paquetes:

  ```
  # apt-get –u dselect-upgrade
  ```

- Para obtener una lista de los paquetes que pueden ser actualizados a nuevas versiones:

  ```
  # apt-show-versions –u
  ```

- Para obtener una lista de paquetes disponibles para instalar cuyo nombre o descripción cuadre con la cadena de texto "mail":

  ```
  $ apt-cache search mail
  ```

- Si se desea obtener información específica sobre el paquete disponible para instalar "apache2" (se mostrará información de todas las versiones disponibles o instaladas):

  ```
  $ apt-cache show apache2
  ```

- Para obtener una lista con los paquetes de que depende "apache2":

  ```
  $ apt-cache depends apache2
  ```

- Para conocer qué paquetes instalados o disponibles para instalar contienen archivos con determinados nombres:

  ```
  $ apt-file archivo
  ```

- También se puede obtener una lista de los archivos que contiene el paquete "manolete" con esta orden:

  ```
  $ apt-file list manolete
  ```

La herramienta *apt* guarda en el archivo /etc/apt/*sources.list* una lista con las fuentes de instalación desde donde se obtienen los paquetes. Este archivo contiene una línea por cada una de las fuentes disponibles y, para cada una de estas líneas, se indican los siguientes campos separados por espacios en blanco:

Tipo de archivos: indica el tipo de archivos que contiene la fuente de instalación: archivos binarios Debian (*deb*), archivos fuente Debian (*debsrc*).

URL : dirección de la fuente de instalación, que puede ser de varios tipos: *http*, *ftp*, *file*, *ssh*, etc.

Argumentos: las opciones indicadas en este campo dependen del tipo de archivos y la distribución utilizada. Por ejemplo, se puede indicar si se trata de un paquete que pertenece a la distribución oficial y con soporte por parte de Ubuntu (*main*) o si el soporte lo ofrece la comunidad (*universe)*. También existen paquetes que no tienen una licencia libre (*restricted*) y los que no tienen ningún tipo de soporte (*multiverse*).

A parte de las herramientas *dpkg* y *apt*, también existen otras basadas en menús y ventanas que simplifican enormemente el trabajo. En Ubuntu se dispone del *Centro de software de Ubuntu* y el *Gestor de paquetes Synaptic*.

1.8. Servicios Web.

1.8.1. El surgimiento de la web.

El éxito de Internet está ligado al desarrollo de la Web de forma que muchas personas no distinguen y confunden los servicios Web con Internet. La realidad actual es que la web ofrece todo tipo de servicios, pero siempre no ha sido así. En sus comienzos Internet no tenía servicios web, sino servicios como el correo electrónico, la transmisión de ficheros (FTP) o los grupos de noticias como Usenet.

La verdad es que estos servicios se manejaban de forma incómoda y sólo los profesionales de la informática podían utilizar Internet. Todo cambio cuando Tim Bernes Lee ideó las páginas web y el éxito de Internet se disparó gracias a la facilidad de manejo que ofreció la web. De tal manera que en poco tiempo los usuarios de las páginas web demandaban que estas ofrecieran más servicios: ya no sólo el mostrar estáticamente un texto fijo, sino que contuvieran vídeo, animaciones,… y poco a poco: acceso a servidores de bases de datos, manejo del correo electrónico, transmisión de ficheros, compra y venta de productos, etc.

Hoy en día desde la Web se puede hacer cualquier tarea, editar documentos, leer el correo electrónico, enviar mensajes, retocar fotos, ver películas, escuchar radio, etc.

1.8.2. Los servicios web.

Hoy en día escuchamos muchas veces en los medios de comunicación la idea de la Web 2.0 para distinguirla de aquella Web que ofrecían los servicios clásicos. En realidad la inmensa mayoría se sitios actuales siguen siendo de este tipo puesto que solo utilizan tecnologías clásicas.

El concepto de Web 2.0 trata de denominar a las páginas web que ofrecen servicios orientados al usuario. Las páginas web tradicionales ofrecían la misma información para todos los usuarios; ahora se personalizan para cada usuario, le permiten un manejo más rico e incluso le hacen partícipe del contenido.

El término Web 2.0 es difícil de definir, pero sin duda hace referencia a un tipo de servicios web muy concretos. Podemos decir que hay tres pilares que conforman las páginas web 2.0:

- **Aplicaciones Ricas de Internet**. Es quizá el término más tecnológico relacionado con la Web 2.0. Se llaman Aplicaciones Ricas de Internet o RIA (*Rich Internet Applications*) a aquellas páginas web que ofrecen servicios que las asemejan con las aplicaciones de escritorio.

  Las RIA permiten utilizar la web como si estuviéramos utilizando una aplicación con toda su potencia en un ordenador local. Inicialmente estas aplicaciones web requerían que el usuario instalara *plugins* (como *JVM* o *Flash* por ejemplo) para poder utilizarlas; pero hoy en día con *AJAX*, *Silverlight* o *HTML 5* basta con el propio navegador.

- **Arquitectura orientada al servicio** o SOA (*Service Oriented Architecture*). Se trata de una tecnología que permite diseñar aplicaciones basándose en peticiones a un determinado servicio. De esta forma se puede crear pequeños componentes software muy reutilizables y además independientes del lenguaje con el que fueron creados.

- **Web social**. Es la parte más evidente y entendible de la Web 2.0. El término se refiere a que el usuario posee una interacción mucho mayor en la web siendo partícipe de lo que en ella ocurre. La Web social está integrada por la llamada comunidad virtual empleando servicios tales como P2P, correo electrónico, redes sociales, blogs, podcast, webcast, wikis, marcadores sociales, sindicación de contenidos RSS, imágenes, video digital. El uso de este entorno ha democratizado el uso de servicios y aplicaciones permitiendo a los usuarios la posibilidad de crearlos, utilizarlos, compartirlos y distribuirlos; es decir es el punto de encuentro en la web social.

En definitiva las páginas Web 2.0 tienen estas características:

- El contenido no solo es creado por los propietarios de las mismas, sino que es creado y modificado por los propios usuarios.
- Las páginas permiten controles más potentes de manejo por los usuarios.
- Los diferentes servicios ofrecidos por distintas páginas tienen facilidad para interaccionar (podemos colocar un mapa de *Google* fácilmente o acceder a *Facebook* para compartir un detalle de la página,…)
- El contenido no está centralizado sino que residen en una maraña (nube) de múltiples servidores colocados incluso en diferentes puntos geográficos.

1.8.3. Aplicaciones en la “nube”.

Es otro de los términos fundamentales para definir los servicios ofrecidos por las páginas actuales. Se basa en ofrecer servicios de modo que el usuario pueda acceder a ellos desde cualquier dispositivo conectado a Internet, ofreciendo una altísima disponibilidad del mismo.

Se basa en la programación distribuida de aplicaciones, pero a tal nivel que las aplicaciones se distribuyen incluso por cientos de servidores de partes distintas del planeta y así poder responder a una demanda ingente de peticiones de servicio y además poseer una altísima capacidad de tolerar fallos.

Esta tecnología es la que permite utilizar el software como un servicio (SaaS, *Software as a Service*), Al usuario le basta un navegador o una pequeña aplicación (como una App de un dispositivo móvil) para acceder y utilizar el servicio.

La computación en la nube apareció para responder tecnológicamente a los retos de empresas como *Google* o *Facebook* que tenían que responder a gran velocidad a las peticiones de millones de usuario. Esto permite que se utilice Internet como la base de trabajo, sustituyendo así al propio ordenador personal al delegar en la *nube* el proceso de las tareas y el almacenamiento de la información.

Ejemplos de servicios en la nube serían:

- Discos duros virtuales. Como *DropBox* o *SugarSync* que sirven para almacenar todo tipo de datos y que estos estén accesibles desde cualquier dispositivo conectado.
- Aplicaciones de ofimática web. Con capacidad para crear e incluso compartir documentos de texto, hojas de cálculo, presentaciones, etc. Entre ellas están *Google Drive* y *SkyDrive*.
- Copias de seguridad en línea. Al estilo del disco virtual, pero pensado para que las empresas tengan un respaldo en caso de pérdida de información.
- Calendarios. Permiten disponibilidad permanente de los datos de agenda personal.
- Sistemas operativos web. Permiten utilizan un ordenador virtual disponible a través de Internet. El más famoso es *EyeOS*, y en España *TribalOS*.
- Servicios *VPN*. Permiten a través de un navegador u otra aplicación de cliente ligera, controlar uno o más ordenadores remoto de modo que no sea necesario acceder físicamente a él.
- Redes sociales. Como *Facebook*, *Tuenti* o *LinkedIn*.
- Bibliotecas multimedia. Como *Youtube* o *Flickr*.
- Marcadores en línea. Como *Delicious* (la antigua *del.icio.us*) o *Digg*.

1.9. Servidores de aplicaciones web.

1.9.1. Servidores web.

Los servidores web son los encargados de recibir las peticiones referidas a páginas o elementos de la web a través del protocolo HTTP. Normalmente es un software alojado en un ordenador que actúa como servidor y a la escucha de las peticiones de los clientes.

Casi siempre es el navegador el que pide al servidor web el recurso que desea el usuario, para finalmente recibir dicho recurso (si fue válida la petición) y traducirle si es necesario a una forma legible por el usuario (es decir la traducción de la respuesta en HTML la hace el navegador) o un mensaje de error si por algún motivo no se ha podido dar respuesta a la solicitud del cliente.

1.9.2. Servidores de aplicaciones

Un servidor de aplicaciones es por así decirlo una ampliación a los servidores web. Es decir, son servidores web, pero que tienen capacidad de almacenar y gestionar aplicaciones web; entendiendo que una aplicación web es un servicio al que los usuarios acceden a través de la web.

Este tipo de servidores no sólo sirven para atender peticiones HTTP, sino que además entienden instrucciones de lenguajes avanzados de la web, o bien son capaces de acceder a recursos de otros servidores. Ese proceso se hace de forma transparente al usuario, es decir el usuario pide el servicio normalmente a través de su navegador, y el servidor de aplicaciones atiende la petición e interpreta el código de la aplicación a fin de traducir y mostrar al usuario el resultado de forma entendible por su navegador (es decir en formato HTML).

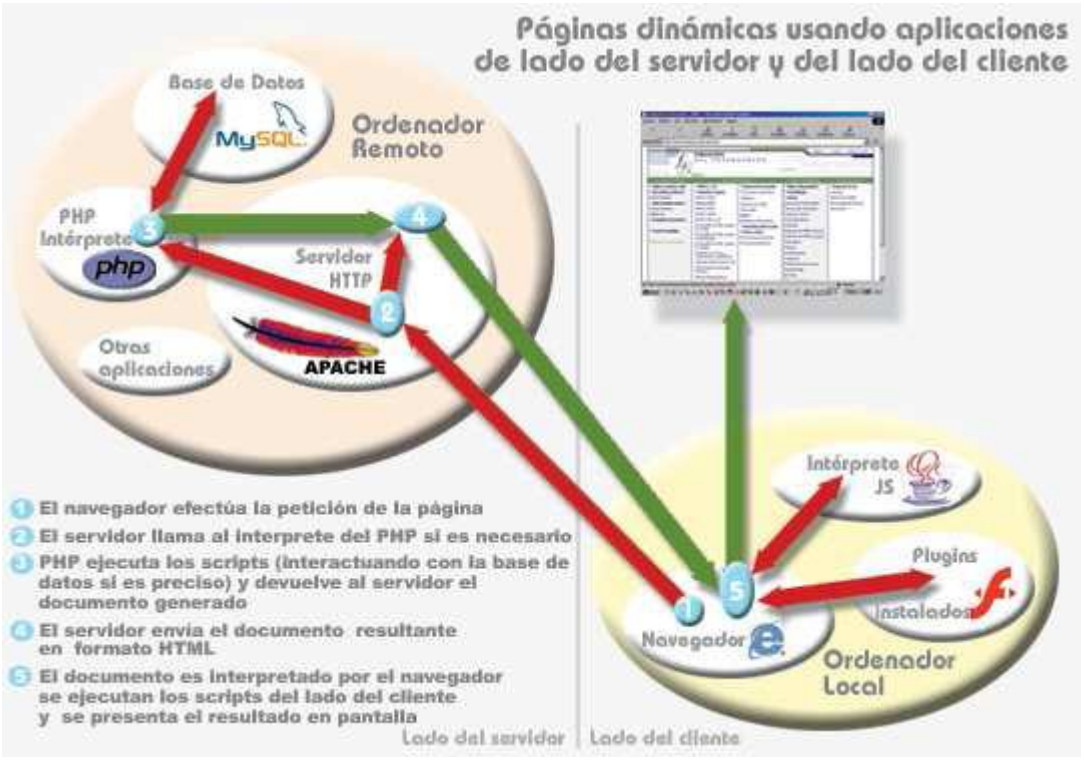
La forma más común de trabajar de un servidor de aplicaciones, es la que se conoce normalmente como arquitectura de tres capas. Otras aplicaciones web se diseñan para trabajar con más capas.

Una primera capa es la del navegador, el cual es capaz de traducir código del lado del cliente (*HTML, JavaScript, CSS, Flash*,…). Para ello dicha capa debe disponer de todos los componentes necesarios para hacer esa labor en el ordenador del usuario.

La segunda capa la forma el servidor de aplicaciones en su labor de traducir código en el lado del servidor (*JSP, PHP, ASP, Ruby on Rails, Cold Fussion*…) y convertirlo al formato entendible por el navegador.

La tercera capa son todos los servicios a los que accede el servidor de aplicaciones para poder realizar la tarea encomendada a la aplicación (por ejemplo el acceso a una base de datos o a otro servidor).

En el siguiente esquema podemos ver los pasos realizados por un cliente cuando solicita una página web dinámica a un servidor de aplicaciones.



1.9.3. Tecnologías de las aplicaciones web.

1.9.3.1 Lenguajes de script de servidor.

- PHP ***(Personal Home Pages)***. Se trata de un lenguaje de scripts de servidor; es decir código que se incrusta en las páginas HTML y que requiere ser traducido por un servidor de aplicaciones que devolverá un resultado en formato HTML.
- ASP ***(Active Server Pages)***. Tecnología de *Microsoft* similar a la anterior, sólo que está pensada para utilizar en servidores de Windows, especialmente en IIS (*Internet Information Sever*) usando como base los lenguajes C# o *VisualBasic.Net*
- JSP ***(Java Server Pages)***. Competidor de ASP desarrollado por *Sun* (ahora *Oracle*) que usa como base el lenguaje *Java*.
- ColdFussion. Otro lenguaje de scripts, propiedad de *Adobe*. Es el más sencillo de todos, pero es de uso más caro porque requiere servidores especiales (Servidores de ColdFussion).

1.9.3.2 Plataformas de desarrollo de servidores web empresariales.

- JavaEE *(Java Enterprise Edition)*. Nombre de la plataforma de creación de aplicaciones web empresariales de Java. Está formada fundamentalmente por el propio lenguaje Java, *EJB* (*Enterprise Java Beans*, componentes reutilizables empresariales), *servlets* y *JSP* además de otros componentes.
- .NET. Plataforma de *Microsoft* que permite (entre otras muchas posibilidades) crear aplicaciones y servicios web, haciendo especial énfasis en el transporte de datos mediante XML.

1.9.3.3 Frameworks MVC.

En inglés *framework* se puede traducir como estructura, aunque otras acepciones del término serían *patrón de diseño* o *plantilla*. MVC son las siglas de *Modelo-Vista-Controlador* un paradigma de programación de aplicaciones que separa en tres niveles el trabajo de la aplicación:

- El **modelo** hace referencia a llamada *lógica de negocio* (tareas relacionadas con los procesos de un negocio, es decir procesamiento de datos, operaciones de cálculo, obtención de resultados, etc.). Normalmente manipula los datos a través de un Sistema Gestor de la Base de Datos.
- La **vista** hace referencia al aspecto visual de la aplicación que se presenta al usuario.

- El **controlador** es la parte que controla los eventos y las acciones del usuario y las comunica a los dos apartados anteriores.

El *framework* MVC, es un modelo de trabajo que facilita la creación de aplicaciones web complejas. Este patrón de diseño se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.

A continuación se expone una lista de los *frameworks* con más éxito, aunque actualmente hay varias decenas en el mercado.

- *Ruby on Rails*. Se trata de un marco de trabajo muy exitoso por la facilidad que tiene de programar y sus buenos resultados visuales. Se puede ejecutar en casi cualquier servidor web, basta con instalar el componente correspondiente. Está desarrollado en lenguaje *Ruby* bajo licencia MIT.
- *Apache Struts*. El marco de trabajo más famoso para la creación de aplicaciones *JavaEE*. Muy preparado para utilizar con *Apache*. Desarrollado en *Java* y con licencia *Apache*,
- *Spring*. Otro marco desarrollado en *Java* y con licencia *Apache* para trabajar en *JavaEE* que tiene bastante éxito. Tiene incluso una versión para las aplicaciones *.NET* (*Spring .NET*).
- *Django*. Escrito en *Python* y con licencia *BSD*, está pensado para utilizar este lenguaje facilitando la creación de aplicaciones web.
- *Zend*. Framework escrito para *PHP* y con licencia *BSD*, es uno de los más populares para este lenguaje.
- *Yii*. Otro framework *PHP* de reciente creación con licencia *BSD* y de gran crecimiento comercial.

1.9.4. Servidores web.

1.9.4.1 Apache.

Apache es el servidor web multiplataforma más popular de la actualidad, abarcando una gran porcentaje de todos los servidores web instalados. Se trata de un software de código abierto multiplataforma que se puede distribuir sin problemas e incluso mejorar. Dispone de multitud de módulos que lo convierten en un servidor capaz de gestionar todo tipo de aplicaciones, lo que también le convierte en el servidor web más popular de la actualidad;

Por ejemplo dispone de módulos para:

- Implementar *SSL*. Protocolo de seguridad en la transferencia de información.
- Enlace con el servidor de aplicaciones *Tomcat*, para implementar aplicaciones web empresariales *Java* de servidor.
- Dispone de módulos para lenguajes de servidor como *Perl*, *PHP*, *Python*, *Ruby*.
- Módulos para soporte con el protocolo WebDAV, autentificación con servidores LDAP, etc.

1.9.4.2 IIS.

Abreviatura de *Internet Information Server*, es el servidor de aplicaciones de Microsoft que está presente en las versiones profesionales de Windows y en todas las de servidor. Viene con el propio sistema operativo y para instalarle basta con agregar componentes al mismo y elegir las funciones deseadas del servidor IIS.

IIS incluye un servidor web (tanto HTTP como HTTPS), servidor FTP, WebDAV y SMTP. Además se comporta como servidor de aplicaciones web desarrolladas con la plataforma *.NET* y admite extensiones para diversos tipos de aplicaciones (incluido *PHP*). Es el máximo competidor de *Apache*.

1.9.4.3 Nginx.

Servidor web multiplataforma de código abierto (licencia *BSD*) cada vez más popular. Además es un servidor *Proxy* para correo electrónico (realiza exploración de entrada/salida de mensajes de correo electrónico en busca de Virus, Gusanos, Troyanos, spam y archivos adjuntos peligrosos o definidos como innecesarios). Se puede ampliar con módulos que permiten servir aplicaciones web. Actualmente empieza a codearse con Apache y IIS.

1.9.5. Servidores de aplicaciones web.

1.9.5.1 Apache Tomcat.

Basado en Apache, y desarrollado por la fundación Apache Software Foundation (http://www.apache.org/); es un servidor web con capacidad para desarrollar aplicaciones *JavaEE*, siendo dentro del software libre el más popular para este tipo de aplicaciones.

Realmente es un contenedor web con soporte de *servlets* que es capaz de compilar JSPs convirtiéndolas en *servlets*. El motor de servlets de Tomcat a menudo se presenta en combinación con el servidor web Apache.

1.9.5.2 WepSphere.

Servidor de aplicaciones web y producto estrella dentro de la familia WebSphere de IBM. (http://www-01.ibm.com/software/websphere/). Construido usando estándares abiertos tales como JavaEE, XML, y Servicios Web. Funciona con varios servidores web incluyendo Apache HTTP Server, Netscape Enterprise Server o Microsoft Internet Information Services (IIS).

1.9.5.3 WebLogic.

Es un servidor de aplicaciones JavaEE y también un servidor web HTTP desarrollado inicialmente por BEA Systems y posteriormente adquirida por la compañía Oracle Corporation. (http://www.oracle.com/technetwork/middleware/weblogic/overview/index.html). Se ejecuta en Unix, Linux, Microsoft Windows, y otras plataformas.