PROGRAMACIÓN DE SERVICIOS Y PROCESOS

TÉCNICO EN DESARROLLO DE APLICACIONES MULTIPLATAFORMA

Protocolos nivel de aplicación

Vamos a recordar qué es un protocolo y vamos a estudiar **algunos de los protocolos** que se usan en comunicaciones y que actúan en la **capa de aplicación**:

- **DNS,**
- **FTP,**
- **SMTP,**
- **POP3,**
- **IMAP,**
- **HTTP**.

Vamos a analizar qué son y para qué sirve cada uno de ellos.

Servicios en red

Para programar aplicaciones que se comuniquen mediante el envío y recepción de mensajes por red, necesitaremos una red de **ordenadores interconectados** entre sí. Esta la definimos formalmente como un **sistema de telecomunicaciones interconectado** entre sí y que tienen la finalidad de **compartir información** o recursos.

Aprovechando esta distribución de datos, se podrá hacer un mejor uso de ellos por parte de todos los usuarios que se encuentren dentro de la red informática, es decir, se va a mejorar, en gran medida, el rendimiento y la eficacia del sistema.

La utilización de **redes de ordenadores** (servicios en red) presenta las siguientes **ventajas**:

- Reduce los **costes** tanto en hardware como en software.
- Podremos crear **grupos de trabajo**, los cuales nos ayudarán a tener organizados a los usuarios.
- Mejora de forma notable la **administración**, tanto de los equipos físicos como de las aplicaciones.
- Se mejora la **integridad** de los datos que hay en el sistema, que será mayor cuantos **más nodos** tenga el mismo.
- Lo mismo para la **seguridad** en los datos.
- Se facilita la **comunicación**.

Los **servicios en red se van a clasificar en**:

- Servicios de administración/configuración:

  Gracias a estos servicios, se va a facilitar tanto la **administración** como la **gestión** de **configuración de los equipos**.

  Un ejemplo son los servicios **DHCP y DNS**.

- Servicios de acceso remoto:

  Con ellos, podremos gestionar las **conexiones de los usuarios a nuestra red** desde lugares remotos.

  Un ejemplo son los servicios **Telnet y SSH**.

- Servicios de ficheros:

  Gracias a estos servicios, podremos ofrecer grandes cantidades de **almacenamiento**. Un ejemplo es el servicio **FTP**.

- Servicios de impresión:

  Nos facilitan la opción de **imprimir documentos** de forma **remota**.

- Servicios de información:

Nos permiten **obtener ficheros en función de** su **contenido**.

Un ejemplo es el servicio **HTTP**.

- Servicios de comunicación:

  Gracias a estos servicios, los usuarios podrán comunicarse a través de **mensajes**.

  Un ejemplo es el servicio **SMTP**.

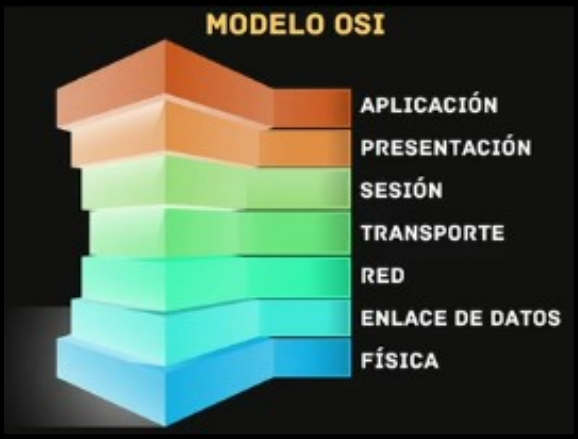
Definición de servicio en red

*Programa o **software** que va a proporcionar una determinada **funcionalidad a nuestro sistema**. Están basados por norma en una serie de **protocolos y estándares**,*

Protocolos a nivel de aplicación

Actualmente, el conjunto de **protocolos más importante** lo conforman los englobados en el modelo **TCP/IP**. Sobre este modelo están **basadas todas** y cada una de las **comunicaciones** que realizamos cotidianamente.

Vamos a destacar, de entre todas las capas que conforman el modelo TCP/IP, la **capa de Aplicación**, que, si recordamos, es la **capa** que está **más arriba** dentro de este modelo. La capa de Aplicación es la que contiene absolutamente todos los **protocolos** que van a estar **relacionados con los servicios en red**.



En la capa de transporte donde, utilizando el protocolo TCP o UDP, enviaremos los datos al destinatario.

En esta capa, se van a **definir los protocolos** que se van a utilizar por todas las **aplicaciones** que desarrollemos que necesiten **intercambiar datos**.

Cabe destacar que el **número de protocolos** que se encuentra en la capa de Aplicación **aumenta** continuamente, ya que no dejan de aparecer **nuevos servicios** que **demandan los usuarios** y que deben basarse en algún protocolo.

Algunos de los **protocolos más importantes** que nos vamos a encontrar dentro de la capa de Aplicación del modelo TCP/IP son los siguientes:

- FTP: Este protocolo se encarga de la transferencia de **ficheros**. Su nombre lo conforman las siglas de Protocolo de Transferencia de Ficheros (**File Transfer Protocol**).
- SMTP: Mediante el que podremos enviar **correos** electrónicos.
- HTTP: Es el que nos habilita la **navegación** por Internet. Es el usado por todos los navegadores **web**.
- Telnet: Este protocolo nos permitirá poder acceder de forma **remota** a un **ordenador** y **manejarlo**.

• SSH: Nos va a permitir una **gestión segura** de forma **remota** de otro ordenador. Es la **versión mejorada del Telnet**.

• NNTP: Con él podremos realizar el envío de **noticias** por la red. Su nombre lo conforman las siglas de Protocolo de Transferencia de Noticias (**Network News** Transport Protocol).

• IRC: Con el que podremos **chatear** vía Internet. Es el protocolo que siguen los chats de Internet (**Internet Relay Chat** o charla interactiva en Internet).

• DNS: Este protocolo nos permitirá **resolver direcciones de red**.

Cómo comenzar una aplicación que use servicios en red

Aunque sea posible una implementación desde cero utilizando Sockets, esto sería una tarea bastante ardua, ya que estos **protocolos no son sencillos**.

Existen **bibliotecas ya predefinidas en la JDK de Java** que nos van a permitir **trabajar con estos protocolos**, facilitándonos el trabajo, en gran medida.

Un ejemplo puede ser la biblioteca **Proyect Lombok**, la cual **automatiza el código generado de los POJO**.

También es importarte tener en cuenta que, probablemente, podremos encontrar **bibliotecas de terceros** que trabajen con estos protocolos y que, seguramente, serán más sencillas de utilizar.

Con respecto a este tipo de bibliotecas, podemos destacar los **controladores** que nos ofrecen bases de datos como MySQL para poder **conectar nuestros proyectos** a ellas.

El protocolo DNS

Todos los **dispositivos**, ya sean ordenadores, portátiles, smartphones, tablets, videoconsolas, etc., que se puedan **conectar a una red TCP/IP** se van a **identificar mediante una dirección IP** del estilo 192.168.1.1.

Las direcciones IP que se utilizan **normalmente** son del formato **IPv4**, las cuales se componen de **cuatro números**, que estarán comprendidos en el intervalo de **0 a 255**, separados por un **punto** unos de los otros.

Los **ordenadores** pueden **utilizar estos valores** muy fácilmente para **comunicarse** entre los diferentes **nodos**, aunque para los humanos puede ser bastante complicado acordarse de todas las IP de los nodos o hosts a los que queremos acceder.

El servicio **DNS**, cuyas siglas hacen referencia a Sistema de Nombre de Dominio (**Domain Name System**), se encarga de resolver el problema de identificar a un nodo mediante una dirección IP. Este servicio nos va a permitir **asignar nombres de dominio a un nodo**, por ejemplo www.google.es, lo cual, para los humanos, es más fácil de recordar.

El servicio DNS se podrá utilizar en **cualquier dispositivo** que se conecte a la **red**.

El objeto principal del servicio DNS es el de **traducir las direcciones IP a nombres de dominio** y **viceversa** en cada uno de los dispositivos que están conectados a la red, pudiendo identificarlos de una forma mucho más sencilla.

Esto no quiere decir que tengamos que hacerlo así obligatoriamente. Si queremos **acceder** a los nodos de nuestra red **mediante direcciones IP**, **podremos** hacerlo sin ningún problema.

Ventajas que nos ofrece el servicio DNS:

- Posibilita que **varios nombres de dominio** compartan una **misma dirección IP** (por ejemplo, si se tienen distintas marcas comerciales y se desarrollan distintas webs, pero se desea que todas apunten al mismo servidor).
- Permitirá que **varias IP** estén compartidas **por varios nombres de dominio**.
- Un **nodo** de la red **podrá cambiar de nombre de dominio** sin tener que cambiar de dirección IP.

El servidor **DNS** va a utilizar una **base de datos distribuida**, que es la encargada de **almacenar** toda la información asociada a **nombres de dominio** en redes como Internet.

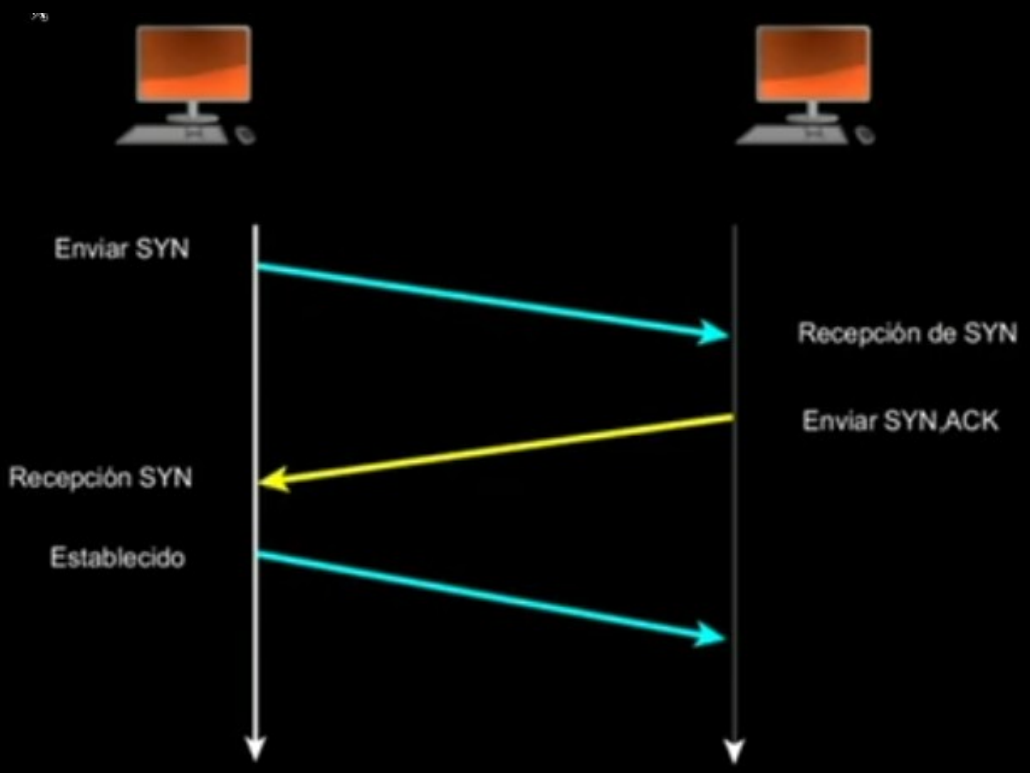
Conexión, comunicación y desconexión

La información pasa de capa en capa desde la capa Aplicación **hasta llegar a la capa de Transporte**, donde mediante una API que utilice sockets se podrá enviar la información.

Un socket es la representación que nos va a dar **java** para **realizar una transmisión** de información en red. En otros lenguajes de programación tendremos otras representaciones equivalentes a sockets.

Los **pasos para crear una comunicación** en red **mediante sockets** son:

1. Se **crean los sockets** tanto en cliente como en servidor.
2. El **servidor** establece un **puerto de escucha**.
3. El **cliente se conecta** al servidor mediante el **puerto** de escucha.
4. El **servidor acepta la conexión** del cliente.
5. Se crean los **flujos de datos** y se realiza el **intercambio** de información.
6. Se **cierran las conexiones** tanto del servidor como del cliente.



El protocolo FTP

Hoy en día, una de las funcionalidades más utilizadas en la red es la de poder **transferir ficheros**, sean del tamaño que sean, desde varios ordenadores, en una red local o desde Internet.

Para proporcionarnos este servicio surgió el protocolo FTP, que proporcionará las funcionalidades necesarias para definir un **estándar en la transferencia de ficheros** en las **redes TCP/IP**.

Principios fundamentales del protocolo FTP (son 2):

- Nos va a permitir **intercambiar ficheros** entre ordenadores remotos que estén interconectados mediante una **red**.
- Permite transferir los ficheros a una **velocidad** bastante **alta**.

La gran **desventaja** de este protocolo es que transmite toda la información en **formato de texto plano**. Esto significa que la información transmitida **no está codificada**, por lo que, si alguien realiza una **conexión intermedia**, podrá **obtener los ficheros** transmitidos de un extremo a otro. La transferencia se realiza en texto plano para proporcionar una **mayor velocidad** de transferencia de información, a costa de una **pobre seguridad**.

El problema de la seguridad será solucionado gracias a la **encriptación** de la información a través del **protocolo SFTP**, **Secure File Transfer Protocol** o Protocolo de Transferencia de Ficheros Seguro, usado **junto al protocolo SSH**.

El protocolo FTP usa el **puerto 20** para la transmisión de **datos** y el **21** para la transferencia de **órdenes**.

Algunas de las **características del protocolo FTP** son:

• Permite **conectar usuarios** en remoto al servidor FTP.

• Hay un **límite en el acceso** al sistema de archivos mediante un **sistema de privilegios** de los usuarios.

• Tiene **dos modos de conexión**:

- modo **activo**, con el que habrá **dos conexiones distintas**,
- modo **pasivo**, con el que **no habrá dos conexiones** distintas.

Protocolo SMTP

El servicio de correo electrónico permite tanto enviar como recibir mensajes con o sin archivos de una forma muy rápida a través de Internet.

El servicio de la capa de Aplicación que nos va a permitir usar el **correo electrónico** es el **protocolo SMTP**, cuyas siglas significan Protocolo para Transferencia de Correo Simple (**Simple Mail Transfer Protocol**), el cual sigue el **modelo Cliente/Servidor**. Por lo tanto, deberemos tener un servidor de correo electrónico y un cliente de correo electrónico para poder usarlo correctamente.

El **servidor** de correo electrónico **creará** una serie de **cuentas** para los usuarios, que **tendrán** lo que se denomina un **buzón**, donde estarán **almacenados todos sus correos** correspondientes.

Los **clientes** serán los encargados de **descargar y elaborar los correos** electrónicos.

El protocolo SMTP utiliza el **puerto 25** y es el encargado de realizar el transporte del correo desde la máquina del **usuario hasta el servidor**, que lo almacenará para que el destinatario del mismo pueda acceder a él.

Cuando el destinatario desee ingresar en su cuenta de correo electrónico, tendrá **dos opciones para acceder a los mensajes**:

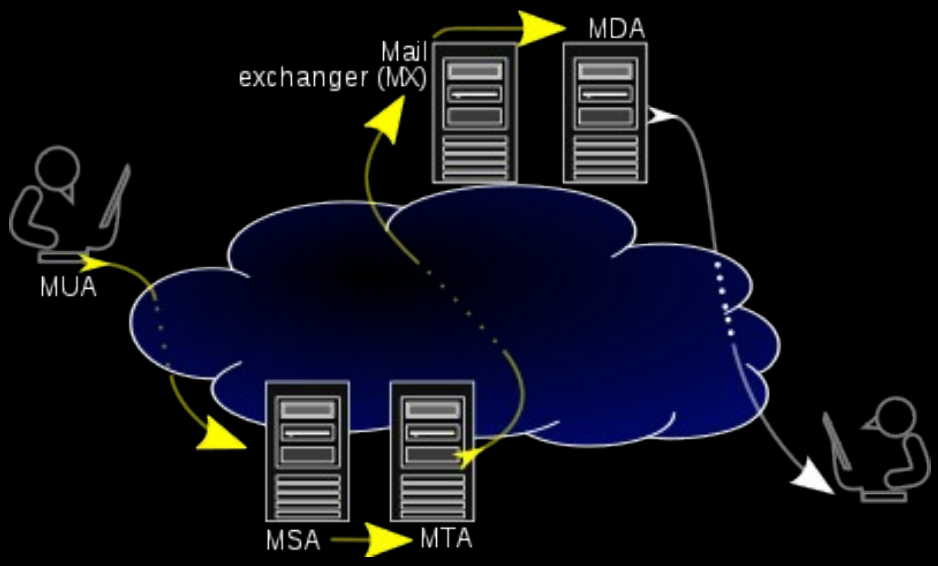
- descargarlos en su **máquina local** mediante el protocolo POP3,
- consultarlos **directamente en el servidor**, mediante el protocolo IMAP.

Estos protocolos nos permitirán no solo enviar texto en nuestros correos electrónicos, sino **cualquier tipo de documento digitalizado**.

Comandos SMTP para realizar transferencia de correo:

- **HELO (hola)**: Comando usado para **abrir una sesión** con el servidor.
- **MAIL FROM**: Comando usado para indicar **quién es el emisor** del correo.

Procesamiento del correo



MUA: agente de usuario de correo.

MSA: agente de sumisión de correo.

MTA (Mail Transfer Agent): Mail Transfer Agent, Agente de Transferencia de Correo.

Para lograr la localización del servidor objetivo, el MTA divisorio tiene que usar el sistema de nombre de dominio (DNS) para lograr la búsqueda del **registro interno de cambiado de correo** conocido como **registro MX** para la esfera del recipiente (la parte de la dirección a la derecha). Es en ese instante cuando el registro de MX devuelto contiene el nombre del anfitrión objetivo.

MDA: agente de entrega de correo.

El protocolo HTTP

El protocolo HTTP, cuyas siglas significan Protocolo de Transferencia de Hipertexto (**HyperText Transfer Protocol**), es el protocolo encargado de que podamos navegar por Internet de forma correcta. Está compuesto por una serie de **normas** que posibilitan una **comunicación y transferencia** de información entre **cliente y servidor**.

La información que transfiere este protocolo son las **páginas HTML**.

El protocolo HTTP se encarga de definir toda la **sintaxis de comunicación** que van a usar tanto el cliente como el servidor, siendo algunas de las **reglas** más importantes las siguientes:

- HTTP sigue el **modelo de petición-respuesta** que aplica al servidor y al cliente.
- El **puerto** que utiliza es el **80**, aunque ofrece la **posibilidad de cambiarlo**.
- Al **cliente** se le denomina como agente del usuario, o **user agent** en inglés.
- Toda la **información que se transmite** se conoce como **recursos** y están identificados mediante una **URL** del tipo **http://**www.google.es.
- Los **recursos también** pueden ser **ficheros**, una consulta de una **base de datos**, un **resultado de una operación** realizada por un programa, **etc**.

El protocolo HTTP **no tiene estado**. Con esto, se refiere a que **no va a recordar** absolutamente nada de **conexiones** anteriores.

Funcionamiento del protocolo HTTP:

- El **usuario accede** a una **URL**, pudiendo **indicar el puerto** en la misma.
- El **equipo cliente descompondrá la información** de la **URL**, diferenciando **todas sus partes**, como el nombre de **dominio**, la **IP**, el **puerto**, etc.
- El **cliente** establece una **conexión con el servidor**.
- El **servidor contesta** al cliente y **envía** el código **HTML**.
- El **cliente visualiza el HTML** y cierra la conexión.