Programación de comunicaciones en red

Programación de servicios y procesos

Objetivos:

Hasta ahora hemos visto como varias aplicaciones pueden colaborar entre sí para realizar una tarea de forma conjunta (**multiproceso**) o bien cómo un mismo programa puede dividir una tarea en partes que se ejecuten de forma concurrente y simultánea (**multihilo**). Todo esto ocurre dentro de una máquina, bien sea en monoprocesador o multiprocesador, controlados por un mismo SO y compartiendo habitualmente parte de la memoria y de la E/S.

En este tema vamos a ir un paso más allá, vamos a crear aplicaciones que funcionen en entornos distribuidos. Volvemos a tener múltiples procesos en ejecución, pero a diferencia de lo que vimos en el tema 2, en el que los procesos tenían una relación padre-hijo (lanzador-lanzado), ahora los procesos se van a ejecutar en sistemas independientes y se comunicarán a través de la red usando **protocolos de comunicación**.

Objetivos de esta unidad:

- Conocer el protocolo TCP/IP, las direcciones usadas en cada capa y los protocolos asociados.
- Conocer las clases que permiten trabajar con direcciones y nombres de servidores.
- Aprender las características básicas de los protocolos TCP y UDP.
- Desarrollar aplicaciones básicas que se comuniquen usando el protocolo TCP.
- Desarrollar aplicaciones básicas que se comuniquen usando el protocolo UDP.
- Diseñar y programar protocolos para la comunicación entre aplicaciones distribuidas.
- Coordinar la ejecución de múltiples clientes en servidores multihilo.

Objetivos:

Procesos e Hilos

Para realizar un programa distribuido en el que se pueda realizar una conexión y una comunicación a través de una red de ordenadores no partimos de cero.

La programación en red está fuertemente ligada a la programación multiproceso. Principalmente en la forma de comunicación que ya vimos entre procesos.

Por otro lado, la especialización y el servicio que ofrece un servidor, de forma simultánea a varios clientes, está basada en la división del trabajo en hilos.

Por todo lo comentado, todos los conceptos y conocimientos adquiridos hasta ahora nos sirven de base para avanzar en los contenidos de este tema.

Contenidos:

1) Conceptos básicos: Comunicación entre aplicaciones
2) Protocolos de comunicación.
3) Puertos de comunicación
4) Modelos de Comunicaciones
5) Sockets

Conceptos básicos.

- Una de las características más apasionantes de la computación es la que tiene que ver con la capacidad de comunicar dispositivos entre sí. Cualquier persona está familiarizada con el uso de aplicaciones que permiten comunicar desde un ordenador hasta un frigorífico con otro dispositivo alojado en cualquier lugar del mundo.

- Debido a esto, no es de extrañar que las empresas realicen y utilicen aplicaciones que se comuniquen por Internet para poder compartir información entre sus empleados. Ejemplo: gestión del stock, facturación, administración de tareas.

- El auge de las tecnologías nos permite realizar estas conexiones mediante pequeños dispositivos como son los portátiles y los teléfonos móviles.

- Antiguamente la programación de aplicaciones que comunican diferentes máquinas era difícil, compleja y fuente de muchos errores; el programador tenía que conocer detalles sobre las capas del protocolo de red incluso sobre el hardware de la máquina. Los diseñadores de las librerías Java han hecho que la programación en red para comunicar distintas máquinas no sea una tarea tan compleja.

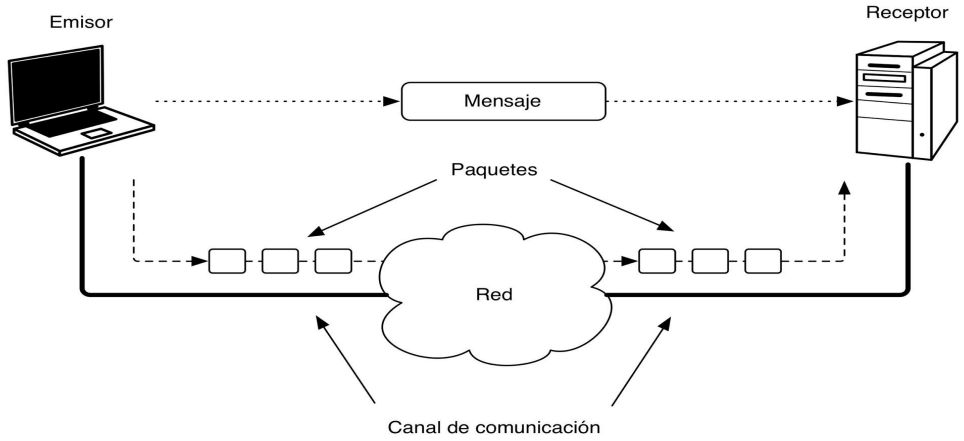
Conceptos básicos.

- Estas aplicaciones siguen un modelo de computación **distribuida** que utilizan el protocolo TCP/IP. Entendemos un sistema de este tipo como:
  - ✔ Está formado por más de un elemento computacional distinto e independiente (un procesador dentro de una máquina, un ordenador dentro de una red, etc), que no comparte memoria con el resto.
  - ✔ Los elementos que forman el sistema distribuido **no están sincronizados** (no comparten un reloj).
  - ✔ Los elementos que forman el sistema están **conectados a una red de comunicaciones**.
- Java dispone de clases para establecer conexiones, crear servidores, enviar y recibir datos, y para el resto de las operaciones utilizadas en las comunicaciones a través de redes de ordenadores. Además, el uso de hilos, que se trataron en el capítulo anterior, nos va a permitir la manipulación simultánea de múltiples conexiones.

Conceptos básicos: Comunicación entre aplicaciones.

- Se asume que el concepto de comunicación implica un **emisor** y un **receptor**. De la misma forma, asumimos que esta comunicación puede ser de 1 a 1 , de 1 a N o de N a M (caso más complejo).

- Además de los elementos anteriores, una comunicación entre aplicaciones, abarcar:
  - ✔ **Mensaje**: Información intercambiada entre las aplicaciones que se comunican.
  - ✔ **Paquete**: Unidad básica de información que intercambian dos dispositivos en una comunicación.
  - ✔ **Canal de comunicación**: Medio por el que se transmiten los paquetes.
  - ✔ **Protocolo de comunicaciones**: Conjunto de reglas que fijan el intercambio de paquetes entre los distintos elementos de la comunicación.



Un poco de historia

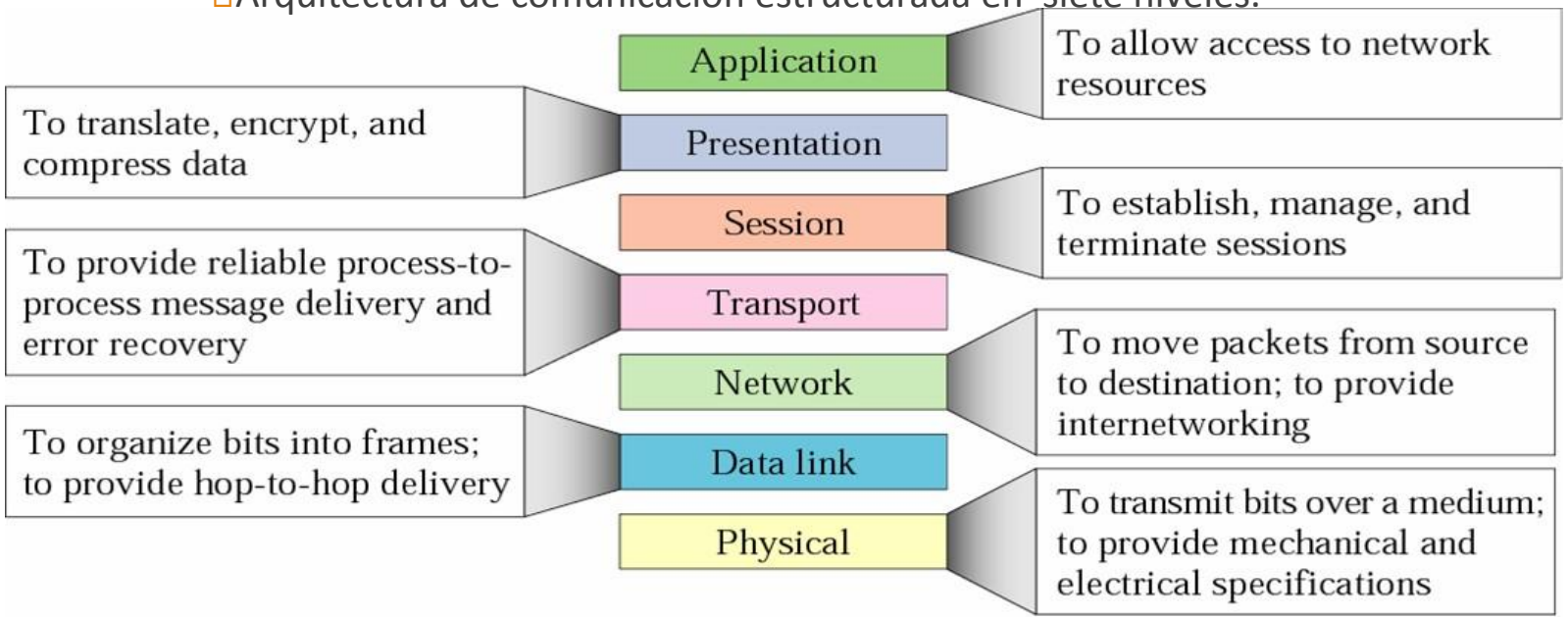
En 1969 la agencia ARPA (Advanced Research Projects Agency) del Departamento de Defensa de los Estados Unidos inició un proyecto de interconexión de ordenadores mediante redes telefónicas. Al ser un proyecto desarrollado por militares en plena guerra fría un principio básico de diseño era que la red debía poder resistir la destrucción de parte de su infraestructura (por ejemplo a causa de un ataque nuclear), de forma que dos nodos cualesquiera pudieran seguir comunicados siempre que hubiera alguna ruta que los uniera.

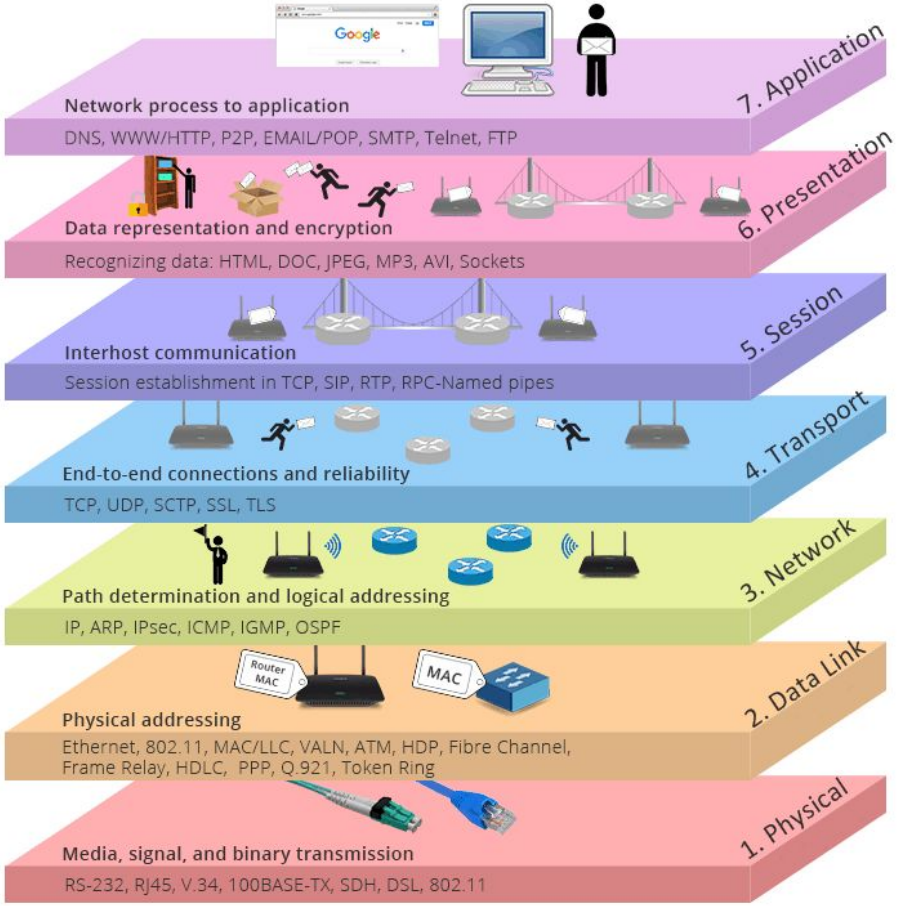
Esto se consiguió en 1972 creando una red de conmutación de paquetes denominada ARPAnet, la primera de este tipo que operó en el mundo. La conmutación de paquetes unida al uso de topologías malladas mediante múltiples líneas punto a punto dió como resultado una red altamente fiable y robusta.

ARPAnet fue creciendo paulatinamente, y pronto se hicieron experimentos utilizando otros medios de transmisión de datos, en particular enlaces por radio y vía satélite; los protocolos existentes tuvieron problemas para interoperar con estas redes, por lo que se diseñó un nuevo conjunto o pila de protocolos, y con ellos una arquitectura. Este nuevo conjunto se denominó **TCP/IP (Transmission Control Protocol/Internet Protocol)**, nombre que provenía de los dos protocolos más importantes que componían la pila; la nueva arquitectura se llamó sencillamente modelo TCP/IP. A la nueva red, que se creó como consecuencia de la fusión de ARPAnet con las redes basadas en otras tecnologías de transmisión, se la denominó Internet.

Modelo OSI y arquitectura TCP/IP

- Modelo de referencia.
- Carácter **teórico**.
- Se propuso para que todas las organizaciones (IBM, Digital, Sun,..) siguieran las **mismas normas** y poder integrar diferentes productos.
- Arquitectura de comunicación estructurada en siete niveles.





Protocolos de comunicación.

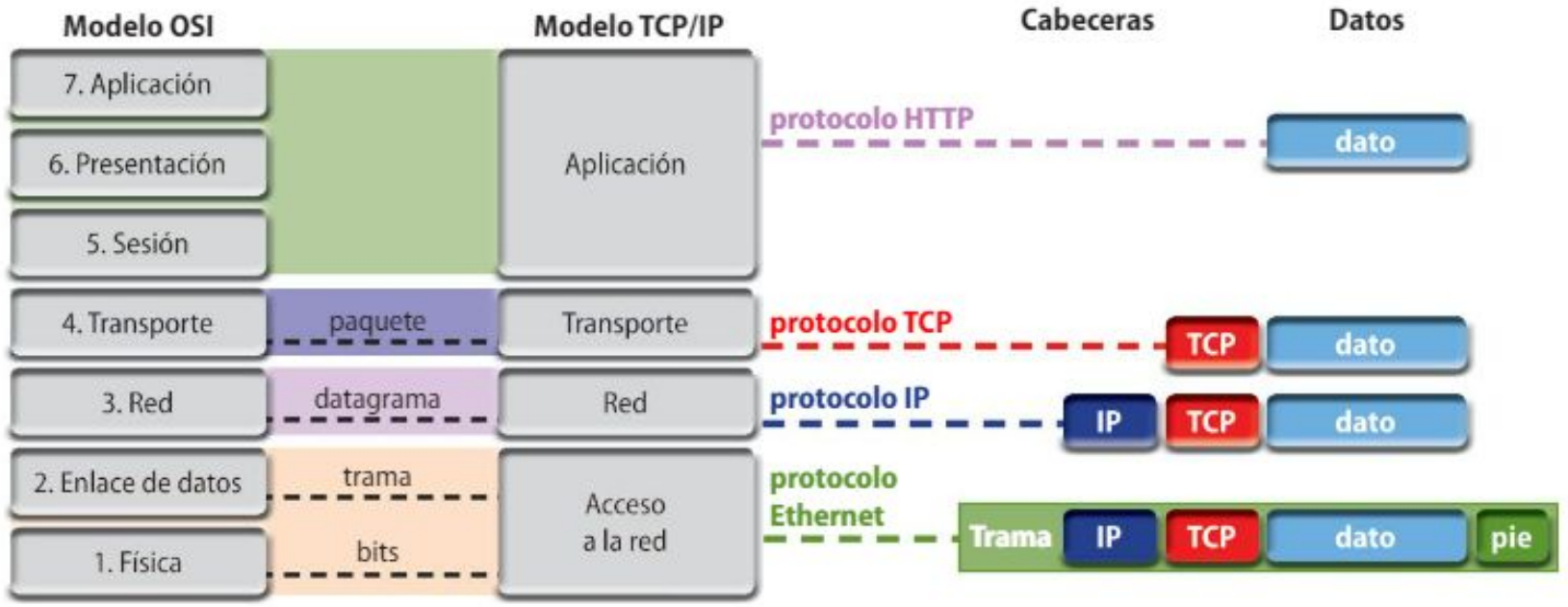
- **TCP/IP** es una familia de protocolos desarrollados para permitir la comunicación entre cualquier par de ordenadores de cualquier red o fabricante, respetando los protocolos de cada red individual. Tiene 4 capas o niveles de abstracción:
  - ✓ **Capa de aplicación:** en este nivel se encuentran las aplicaciones disponibles para los usuarios. Por ejemplo, FTP, SMTP, Telnet, HTTP, etc.
  - ✓ **Capa de transporte:** suministra a las aplicaciones servicio de comunicaciones extremo a extremo utilizando dos tipos de protocolos:
    - Lo componen los elementos software cuya función es crear el canal de comunicación, descomponer el mensaje en paquetes y gestionar su transmisión entre el emisor y el receptor.
    - Los principales protocolos asignados a este nivel son el TCP y el UDP.
  - ✓ **Capa de red:** tiene como propósito seleccionar la mejor ruta para enviar paquetes por la red. El protocolo principal que funciona en esta capa es el Protocolo de Internet (IP)
  - ✓ **Capa de enlace o interfaz de red:** es la interfaz con la red real. Recibe los datagramas de la capa de red y los transmite al hardware de la red.
    - Lo componen los elementos software que se encargan de dirigir los paquetes por la red, asegurándose de que lleguen a su destino.
    - Establece direccionamiento tanto a nivel individual como nivel grupal.
    - La operación básica es el enrutamiento, responsabilidad de los **routers**.

Arquitectura TCP/IP

Estándar para la comunicación en redes de datos.

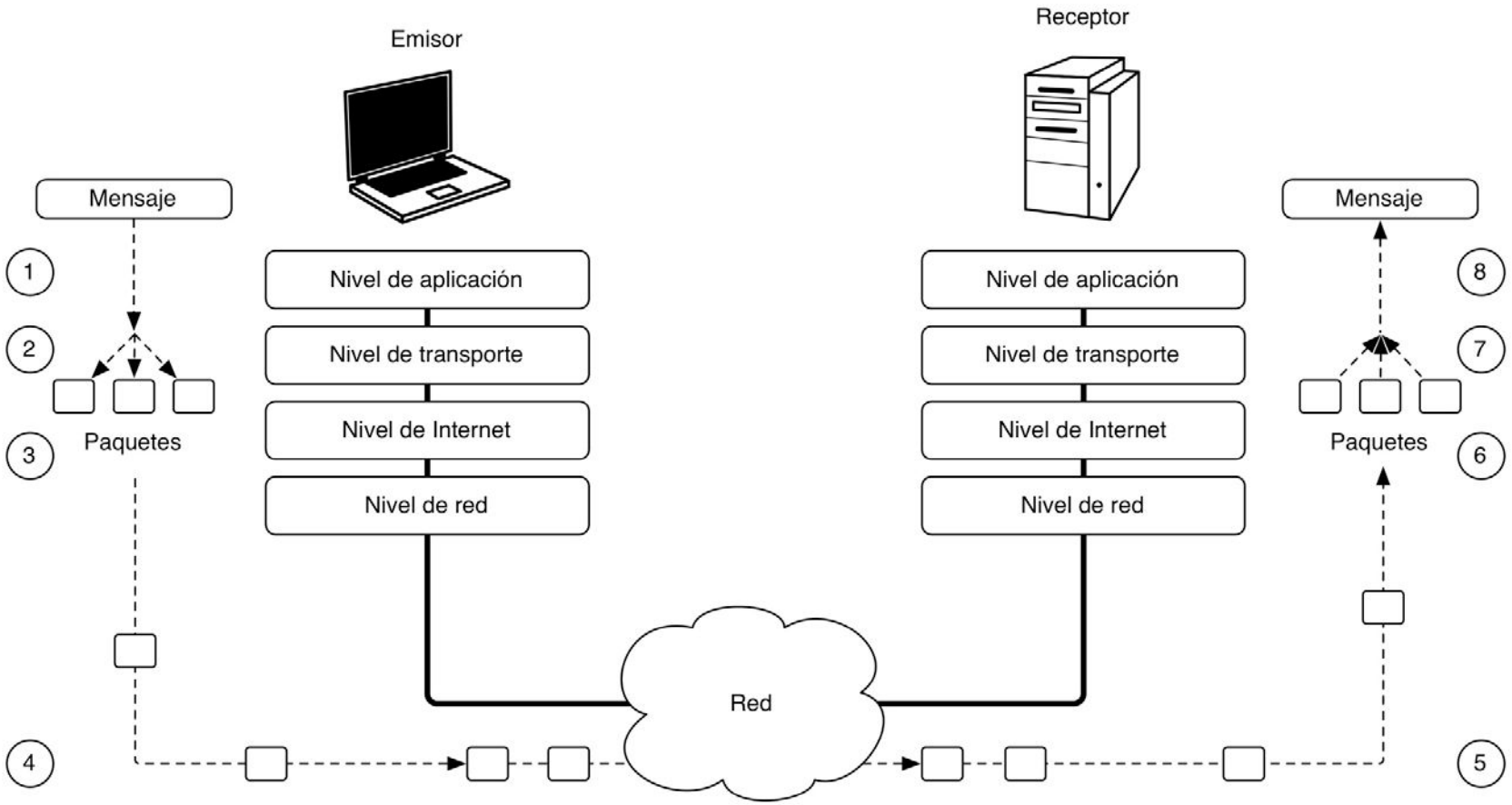
Utilizado para la interconexión de dispositivos a nivel **global**: fundamento de Internet.

Estructura de niveles:



Equivalencia entre los modelos OSI y TCP/IP y transferencia de la información

Arquitectura TCP/IP



Protocolo TCP (Transmission Control Program)

TCP (Transmission Control Protocol): es un protocolo orientado a la conexión que permite que un flujo de bytes originado en una máquina se entregue sin errores en cualquier máquina destino. Este protocolo fragmenta el flujo entrante de bytes en mensajes y pasa cada uno a la capa de red. En el diseño, el proceso TCP receptor reensambla los mensajes recibidos para formar el flujo de salida. TCP también se encarga del control de flujo para asegurar que un emisor rápido no pueda saturar a un receptor lento con más mensajes de los que pueda gestionar.

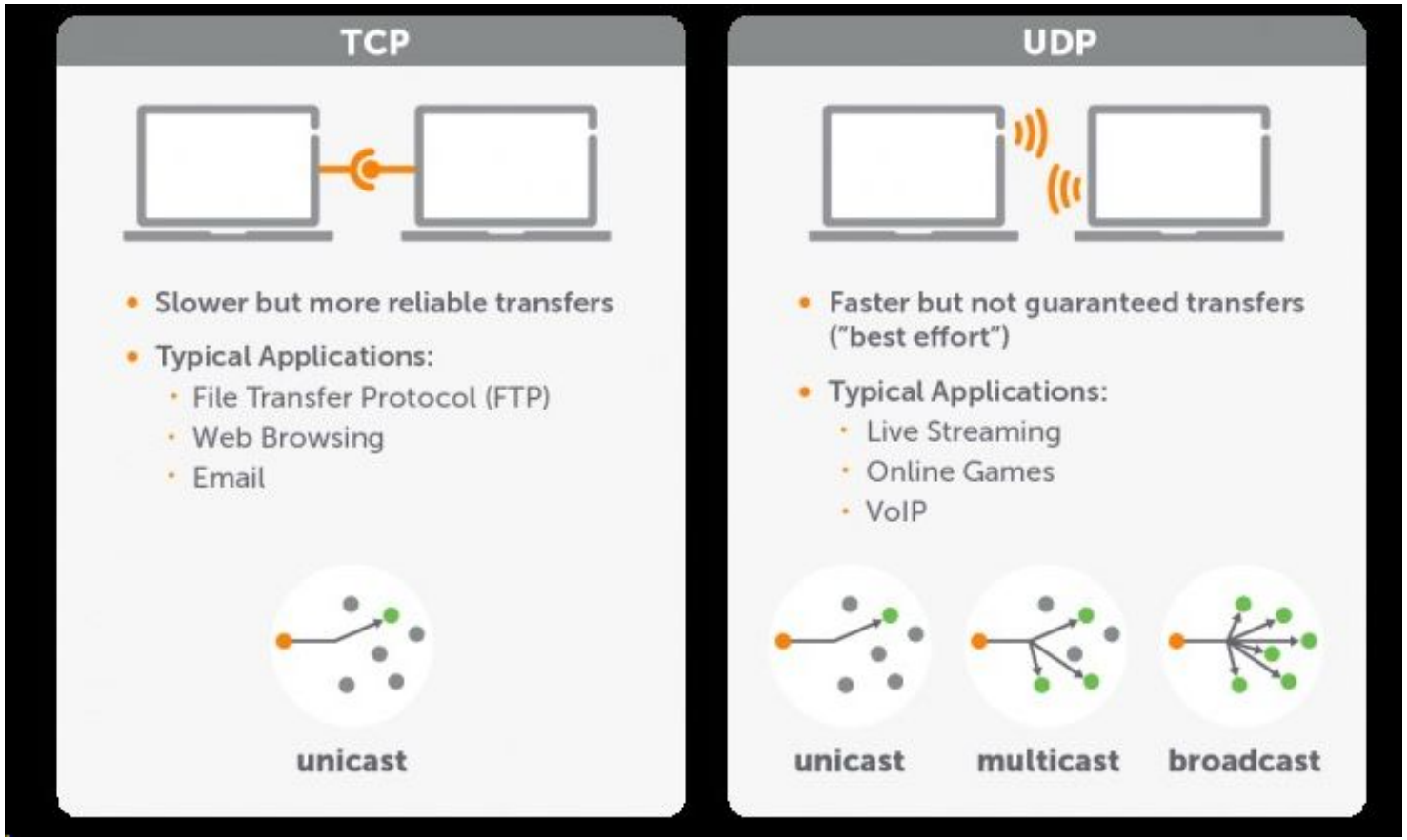
- Garantiza que los datos no se pierden.
- Garantiza que los mensajes llegarán **en orden**.
- Se trata de un **protocolo orientado a conexión**. En el cual, el canal de comunicaciones entre dos aplicaciones permanece abierto durante un cierto tiempo, permitiendo enviar múltiples mensajes de manera fiable por el mismo.
- Este tipo de protocolos opera:
    1. Establecimiento de la conexión.
    2. Envío de mensajes.
    3. Cierre de la conexión.

Protocolo UDP (User Datagram Protocol)

UDP (User Datagram Protocol). Es un protocolo sin conexión, para aplicaciones que no necesitan la asignación de secuencia ni el control de flujo TCP y que desean utilizar los suyos propios. Este protocolo también se utilizan para las consultas de petición y respuesta del tipo cliente-servidor, y en aplicaciones en las que la velocidad es más importante que la entrega precisa, como las transmisiones de voz o de vídeo. Uno de sus usos es en la transmisión de audio y vídeo en tiempo real, donde no es posible realizar retransmisiones por los estrictos requisitos de retardo que se tiene en estos casos.

- **Es un protocolo NO orientado a conexión**. Esto lo hace más rápido que TCP, ya que no es necesario establecer conexiones, etc.
- No garantiza que los mensajes lleguen siempre.
- No garantiza que los mensajes lleguen en el mismo orden que fueron enviados.
- Permite enviar mensajes de 64 KB **como máximo**.
- En UDP, los mensajes se denominan ***datagramas***.

TCP vs UDP



TCP	UDP
Transmission Control Protocol	User Datagram Protocol
Orientado a la conexión	No es orientado a conexión
Confiable	No confiable
Estabelece un Handshake (Negociación)	No hay Handshake
Retransmisión de segmentos y control de flujo	No control de flujo y retransmisiones
Secuenciación de segmentos	No secuenciación
Uso de ACK	No uso de ACK
No soporta Multicasting y Broadcasting	Soporta Multicasting y Broadcasting
Paquetes son llamados Segmento	Paquetes son llamados Datagrama
HTTP, HTTPS, FTP	DNS, Telefonía IP, DHCP

Puertos de comunicación

Con la capa de red se consigue que la información vaya de un equipo origen a un equipo destino a través de su dirección IP. Pero para que una aplicación pueda comunicarse con otra aplicación es necesario establecer a qué aplicación se conectará.

En términos generales, un ordenador tiene una única conexión física a la red. Los datos destinados a este ordenador llegan a través de esa conexión. Sin embargo, los datos pueden estar destinados a diferentes aplicaciones que se ejecutan en el ordenador. Entonces, ¿cómo sabe el ordenador a qué aplicación enviar los datos? Mediante el uso de puertos.

El método que se emplea es el de definir direcciones de transporte en las que los procesos pueden estar a la escucha de solicitudes de conexión. Estos puntos terminales se llaman puertos.

Puertos de comunicación

Aunque muchos de los puertos se asignan de manera arbitraria, ciertos puertos se asignan, por convenio, a ciertas aplicaciones particulares o servicios de carácter universal. De hecho, la IANA (Internet Assigned Numbers Authority) determina, las asignaciones de todos los puertos. Existen tres rangos de puertos establecidos:

- **Puertos conocidos [0, 1023]**. Son puertos reservados a aplicaciones de uso estándar como: 21 – FTP (File Transfer Protocol), 22 – SSH (Secure SHell), 53 – DNS (Servicio de nombres de dominio), 80 – HTTP (Hypertext Transfer Protocol), etc.
- **Puertos registrados [1024, 49151]**. Estos puertos son asignados por IANA para un servicio específico o aplicaciones. Estos puertos pueden ser utilizados por los usuarios libremente.
- **Puertos dinámicos [49152, 65535]**. Este rango de puertos no puede ser registrado y su uso se establece para conexiones temporales entre aplicaciones.

Cuando se desarrolla una aplicación que utilice un puerto de comunicación, optaremos por utilizar puertos comprendidos entre el rango 1024-49151.

Modelos de comunicaciones

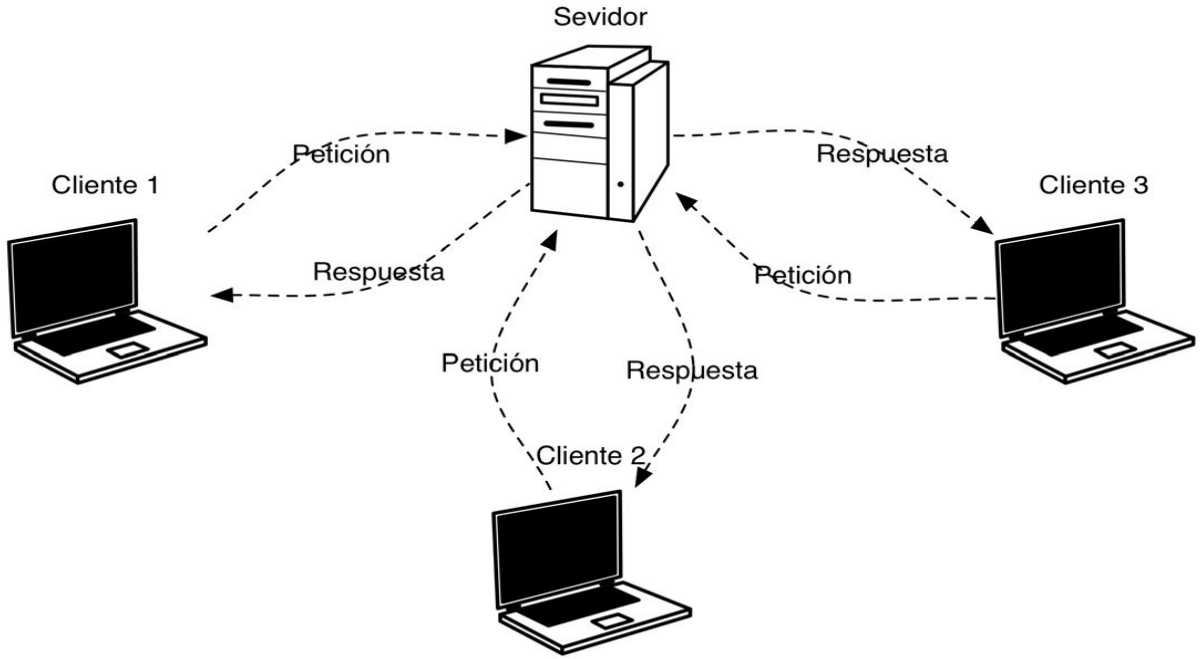
- Un modelo de comunicaciones es una arquitectura general que especifica cómo se comunican entre sí los diferentes elementos de una aplicación distribuida.

- Un modelo de comunicaciones normalmente define aspectos como cuántos elementos tiene el sistema, qué función realiza cada uno, etc.

- Los modelos más usados en la actualidad son:
  - ✔ Cliente/servidor.
  - ✔ Comunicación en grupo.

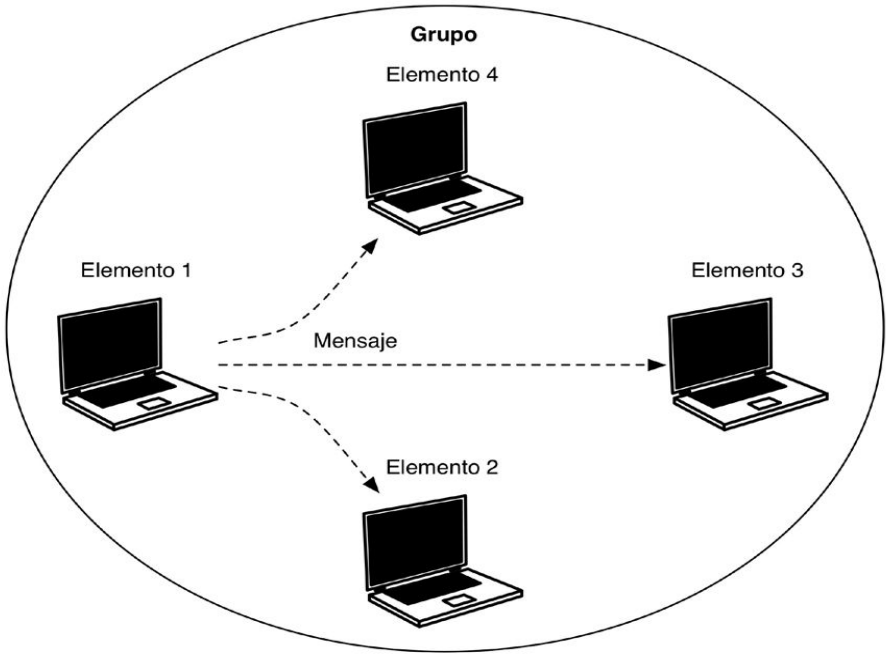
Modelo cliente/servidor

- Es más sencillo de los comúnmente usados en la actualidad.
- En este modelo, un proceso central, llamado *servidor*, ofrece una serie de servicios a uno o más procesos *cliente*.
- El proceso *servidor* debe estar alojado en una máquina fácilmente accesible en la red, y conocida por los *clientes*.
- Cuando un *cliente* requiere sus servicios, se conecta con el *servidor*, iniciando el proceso de comunicación.



Modelo de comunicación en grupo

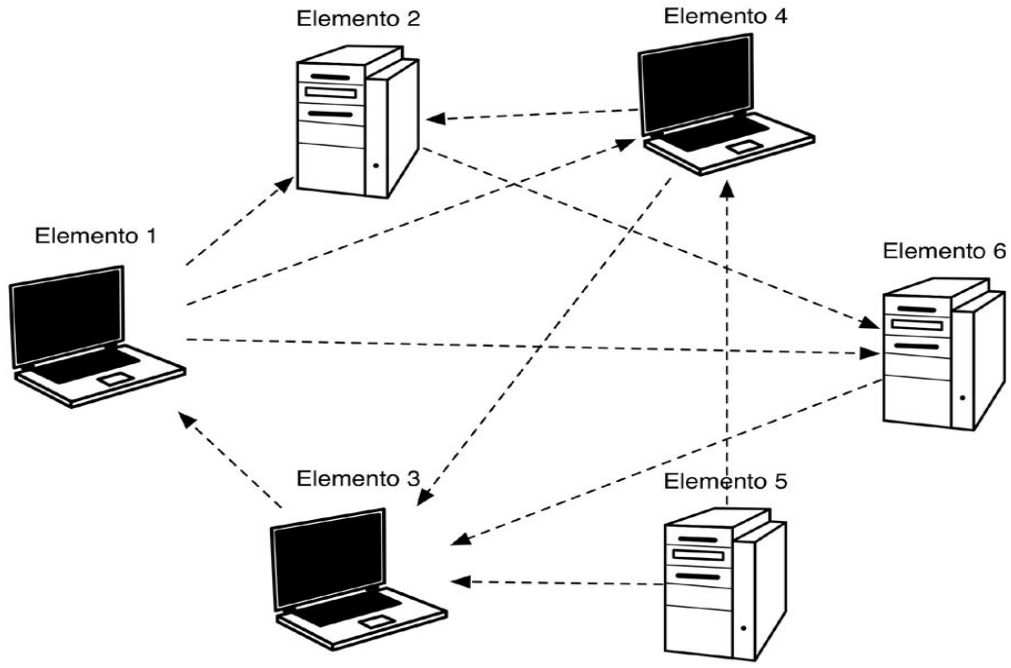
- Es la alternativa más común al modelo cliente/servidor.
- En este modelo no existen roles diferenciados.
- En la comunicación en grupo existe un conjunto de dos o más elementos (procesos, aplicaciones, etc.) que cooperan en un trabajo común.
- A este conjunto se le llama grupo, y los elementos que lo forman se consideran todos iguales, sin roles ni jerarquías definidas.
- Los mensajes se transmiten mediante radiado.



Modelo híbridos

- Las aplicaciones distribuidas más avanzadas suelen tener requisitos de comunicaciones muy complejos, que requieren de modelos de comunicaciones sofisticados.

- En muchos casos, los modelos de comunicaciones reales implementados en estas aplicaciones mezclan conceptos del modelo cliente/servidor y la comunicación en grupo, dando lugar a enfoques híbridos, como las redes **peer-to-peer (P2P).**



Modelo híbridos

- Una **red P2P** está formada por un grupo de elementos distribuidos que colaboran con un objetivo común.
- Cualquier elemento puede desempeñar los roles de servidor o cliente, como si de un modelo cliente/servidor se tratase.
- Las redes P2P puedan ofrecer servicios de forma similar al modelo cliente/servidor.
- Cualquier aplicación puede conectarse a la red como un cliente, localizar un servidor y enviarle una petición.
- Si permanece en la red P2P, con el tiempo ese mismo cliente puede hacer a su vez de servidor para otros elementos de la red.

Sockets

- Los sockets son el mecanismo de comunicación para realizar transferencias de información entre aplicaciones.
- Proporcionan una **abstracción** de la pila de protocolos.
- Un socket representa el extremo de un canal de comunicación establecido entre un emisor y un receptor.
- Su finalidad principal es crear aplicaciones Cliente/Servidor.
- Dado el punto anterior, lógicamente, nos hará falta: una dirección IP y un puerto.
- Los sockets que usan UDP se denominan **datagram sockets**, mientras que los que utilizan TCP se denominan **stream sockets**.