En este módulo se verá:

1. Diferencia entre modelo de referencia y modelo de protocolo.
2. Desarrollo modelo de protocolo TCP/IP.
3. Relación entre cliente y servidor.
4. Modelo de capas para identificar problemas de red

Modelo de capas

Los **modelos en capas** se utilizan para ayudar a **visualizar la interacción entre diversos protocolos.**

Un modelo en capas describe el funcionamiento de los protocolos que se produce en cada capa y la interacción de los protocolos con las capas que se encuentran por encima y por debajo de ellas, describiendo, además, el flujo de información en el proceso de comunicación en uno u otro sentido.

Beneficios

El modelado en capas, más allá de las redes, trae beneficios a la hora de diseñar protocolos, procesos, productos y tecnologías:

Ayuda en el **diseño de protocolos**, ya que los protocolos que operan en una capa específica tienen información definida según la cual actúan, y una interfaz definida para las capas superiores e inferiores.

Fomenta la **competencia**, ya que los productos de distintos proveedores pueden trabajar en conjunto.

Evita que los cambios en la tecnología o en las capacidades de una capa **afecten otras capas** superiores e inferiores.

Proporciona un **lenguaje común** para describir las funciones y capacidades de redes.

Modelo de protocolo

Este modelo describe con precisión la **estructura** de un **conjunto de protocolos** determinado.

El conjunto jerárquico de protocolos relacionados **representa toda la funcionalidad requerida** para todos los elementos, tanto software como hardware, de una red de datos.

El **modelo TCP/IP es un modelo de protocolo,** porque **describe las funciones** que tienen lugar **en cada capa de protocolos** dentro de una **suite TCP/IP.**

Modelo de referencia

Este modelo es coherente con todos los tipos de servicios y protocolos de red al describir **qué es lo que se debe hacer** en una capa determinada, pero no determina la forma en que se debe lograr.

Un modelo de referencia no especifica la forma de implementación ni proporciona detalle suficiente para definir de forma precisa los servicios y protocolos intervinientes de la arquitectura de red.

El objetivo principal de un modelo de referencia es **ayudar a lograr un mejor entendimiento de las funciones y procesos involucrados**.

El **modelo OSI** es el modelo de referencia de internetwork más conocido. Se usa para diseño de redes de datos, especificaciones de funcionamiento y resolución de problemas.

Modelo de capas TCP/IP

El **modelo TCP/IP** es un modelo de protocolo usado para comunicaciones en redes y, como todo protocolo, describe un conjunto de guías generales de operación para permitir que un equipo pueda comunicarse en una red.

TCP/IP provee conectividad de extremo a extremo especificando cómo los datos deberían ser formateados, direccionados, transmitidos, enrutados y recibidos por el destinatario.

Las redes de datos actuales se describen bajo el modelo de protocolo TCP/IP el cual está ampliamente difundido e implementado prácticamente en todas las infraestructuras de redes, tanto redes LAN como Internet.

El **modelo TCP/IP** describe el **proceso de comunicación** y los **protocolos utilizados** y a diferencia del modelo OSI, este posee **4 capas**:

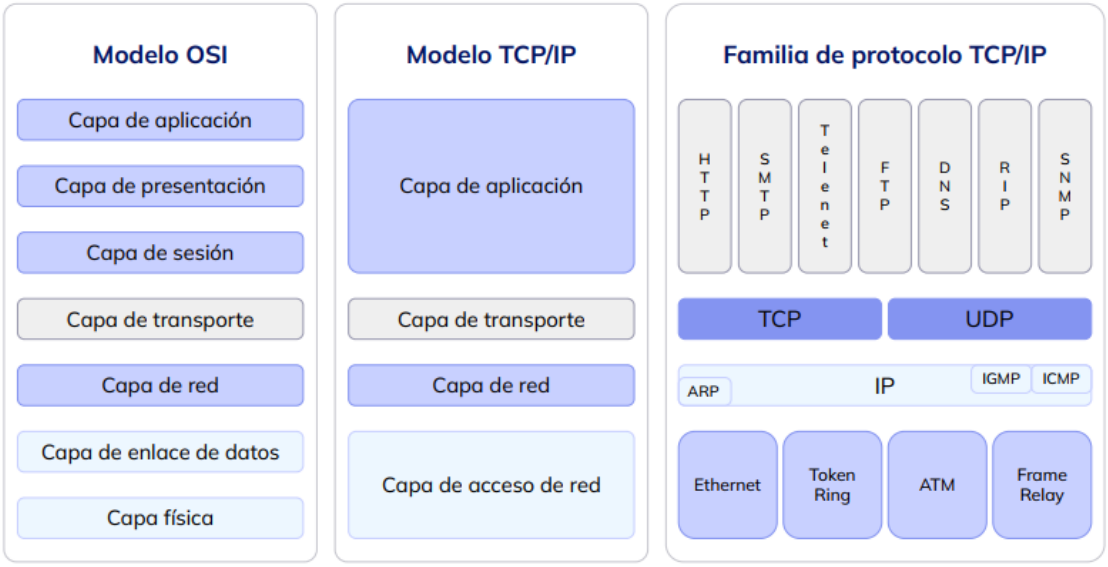
La **capa de aplicación** del modelo TCP/IP es similar a las capas 5, 6, 7 combinadas del modelo OSI, esta capa reúne todo lo que tenga que ver con la generación y formateado del mensaje y cómo se comunican las aplicaciones entre sí.

La **capa de transporte** de TCP/IP abarca las responsabilidades de la capa de transporte OSI e intervienen los protocolos TCP y UDP y algunas de las responsabilidades de la capa de sesión OSI.

La **capa de red** o también llamada ***Internetworking*** utiliza los protocolos de

enrutamiento IP.

La **capa de acceso a la red** del modelo TCP/IP abarca el enlace de datos y las capas físicas del modelo OSI.



Capa 4: Aplicación

Esta capa describe los **protocolos** utilizados por la mayor parte de las aplicaciones para proporcionar servicios de usuario o intercambiar datos de aplicaciones a través de las conexiones de red establecidas por los protocolos de las capas inferiores. Esto puede incluir servicios básicos de soporte de red, como protocolos de enrutamiento y configuración de host.

Algunos ejemplos de esto son el protocolo HTTP o Protocolo de Transferencia de Hipertexto, el protocolo FTP o Protocolo de Transferencia de Archivos, el protocolo SMTP o protocolo de Transferencia de Correo y el Protocolo DHCP o Protocolo de Configuración Dinámica de Host.

La **capa de aplicación** en el modelo **TCP/IP** corresponde a una combinación de la quinta (sesión), sexta (presentación) y séptima capa (aplicación) del modelo OSI.

Capa 3: Transporte

Esta capa se alinea con la capa 4 del modelo OSI.

En la **capa de transporte** se establecen canales de datos mediante puertos de red que es lo que hace posible el intercambio de datos. Además establece la **conectividad de host a host** en forma de servicios de transferencia de mensajes de extremo a extremo independientes de las redes subyacentes e independientes de la estructura de los datos del usuario y la logística del intercambio de información.

Las conexiones entre hosts se basan en 2 tipos de protocolos:

Orientados a conexión bajo el protocolo **TCP** (*Transfer Control Protocol*) con control de errores.

No orientado a la conexión bajo el protocolo **UDP** (*User Datagram Protocol*) sin control de errores.

Los protocolos de esta capa pueden proporcionar control de errores, segmentación, control de flujo, control de congestión y direccionamiento de aplicaciones.

Relación Cliente-Servidor

Las aplicaciones que envían o reciben información de la red de datos tienen roles:

Cliente: un programa que se conecta y transfiere datos contra otra aplicación que puede estar en un host remoto o de forma local.

Servidor: un programa que ofrece un servicio a la red y al que se conectan los clientes.

En una red LAN resulta muy común el compartir archivos entre hosts. En el caso de Windows, basta con elegir una carpeta y compartirla: en el sistema existe un servicio que administra los recursos compartidos del sistema en la red, en este caso un carpeta.

El host que comparte la carpeta asume el rol de servidor, mientras que el resto asumen el rol de clientes, son quienes acceden al servicio ofrecido por el host 192.168.1.100.

Llamaremos **servicios de red** o **servicios** a aquellos programas que ofrecen algún tipo de servicio a la red, por ejemplo, servicio de archivos compartidos de windows, y **servidor** a aquel host que corre dicho servicio, entonces el host 192.168.1.100 es un **servidor de archivos**.

Cualquier tipo de host puede ser un servidor, el rol no está definido por una arquitectura de hardware en particular sino por el tipo de servicios que ejecuta.

Una impresora de red ejecuta un servicio de impresión, es un servidor de impresión, no obstante, se suele dedicar el rol a equipos con arquitecturas específicas que permiten ejecutar servicios y atender gran cantidad de solicitudes y el procesamiento asociado a dicha solicitudes. Estos son hosts diseñados específicamente para ser servidores, ejecutar sistemas operativos específicos y de gran capacidad de cómputo.

Puertos de red

Los servicios reciben las solicitudes de los clientes, que básicamente son mensajes generados en la capa de aplicación, formateados bajo un protocolo en la misma capa y segmentados en la capa de transporte.

Pero para que el servicio pueda recibirlos, estos segmentos tienen que ser entregados por un canal de transmisión específico. La capa de transporte establece el concepto de **puerto de red**. A través de estos puertos se **recibirán las solicitudes** y se procederá al **intercambio de datos**.

Un puerto es una **construcción lógica numerada** que es asignada de forma específica para cada uno de los canales de comunicación que necesita un determinado servicio. Para muchos tipos de servicios, estos números de puerto se han estandarizado para que las computadoras cliente puedan abordar servicios específicos de una computadora servidor sin la participación de servicios de directorio o descubrimiento de servicios. Por otra parte, el cliente también hace uso de puertos para establecer la conexión, entonces la comunicación entre cliente y servidor se hace de puerto a puerto.

Los **servicios** escuchan en **puertos específicos estandarizados**, no importa el tipo software que se utilice para brindar servicios web, siempre lo harán en el **puerto 80**. Para la aplicación **cliente** es indistinto el software utilizado como servidor web, siempre enviará las solicitudes al **puerto específico** para el protocolo asociado, en este caso el protocolo HTTP.

Si bien los servicios escuchan en un puerto específico estandarizado, los clientes suelen usar puertos aleatorios en un rango alto. Existe un número limitado de puertos que es de 65536, esos puertos los podemos clasificar en puertos bajos que ya están definidos para distintos tipos de servicios, mientras que los puertos altos tienen usos variables.

Cabe destacar que aunque los servicios ya tienen puertos asignados es posible **asignar un puerto de escucha distinto** en la configuración del servicio, en estos casos se debe

indicar en el cliente no solo el host al que conectarse sino también al puerto que está fuera del estándar.

En la aplicación se indica la IP del servidor y el puerto de escucha. En Wikipedia podemos obtener una lista de puertos de red reservados y los servicios asociados: Puertos de red.

Protocolo con control de transferencia TCP

TCP es un protocolo orientado a la **conexión** que aborda numerosos problemas de confiabilidad al proporcionar un **flujo de bytes confiable:**

Los datos llegan ordenados.

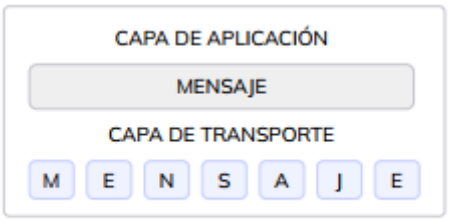
Los datos tienen la cantidad mínima de errores.

No llegan duplicados.

Asegura que los paquetes lleguen a su destino.

Incluye control de congestión de tráfico.

Cuando los mensajes pasan de la capa de aplicación a la de transporte, la cantidad de bytes que forman el mensaje se dividen en segmentos más pequeños. Esto sucede siempre en la capa de transporte, ya sea bajo el protocolo TCP o UDP.



Mensaje segmentado.

Cabecera o Header TCP

A cada segmento se le asigna una **cabecera**, un conjunto de bytes que precede el conjunto de bytes que contiene parte del mensaje.

Esta cabecera contiene información relevante para garantizar la integridad del mensaje, los puertos de origen y puertos destino y estados de conexión, entre otros.

Así como se explicó con las direcciones IP, toda la información en un sistema computacional se expresa en bits, cuando agrupamos estos bits en conjuntos de 8 bits tenemos bytes

Tanto los datos que componen el mensaje como la cabecera TCP son un conjunto de bits que se leen un sentido, cada bit o conjunto de ellos representan algún tipo de información en el segmento, así como en una dirección IP los bits representan los octetos y sus valores numéricos.

Offsets	Octeto	0								1								2								3							
Octeto	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Puerto de origen																Puerto de destino															
4	32	Número de secuencia																															
8	64	Número de acuse de recibo (si ACK es establecido)																															
12	96	Longitud de Cabecera				Reservado				NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Tamaño de Ventana														
16	128	Suma de verificación																Puntero urgente (si URG es establecido)															
20 ...	160 ...	Opciones (si la Longitud de Cabecera >5, relleno al final con "0" bytes si es necesario) ...																															

Cabecera TCP y su organización binaria.

Puerto de origen y destino

Como ya se estableció, los mensajes se pasan de un host a otro mediante puertos. Cuando el host recibe un segmento al leer el campo "**puerto destino**" sabe a qué aplicación debe entregar el mensaje ya que la aplicación está esperando recibir los mensajes por dicho puerto.

Un host recibe segmentos continuamente para distintos tipos de servicios, esta es la manera que tiene de saber y agrupar los segmentos en función de la aplicación a la que va dirigido el segmento.

Octeto	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Puerto de origen																Puerto de destino															

El campo "**puerto de origen**" le dice al host receptor a que puerto destino debe entregar los segmentos de respuesta.

El cliente envía una petición para acceder al contenido web al puerto 8080 del host 192.168.1.200, el servidor recibe los segmentos y elabora los segmentos de respuesta que contienen el recurso solicitado (un index.html, por ejemplo) con el puerto destino 35665.

Control de transferencia: número de secuencia y acuse recibo.

El mensaje se puede dividir en miles de segmentos, estos se envían de uno en uno y de forma secuencial:

La cabecera incluye dos campos:

Número de secuencia: se envían los segmentos en orden secuencial, en el destino se arma la información respetando este orden.

Número de acuse recibo: cuando el receptor recibe un segmento devuelve un segmento con el siguiente valor al número de secuencia recibido.

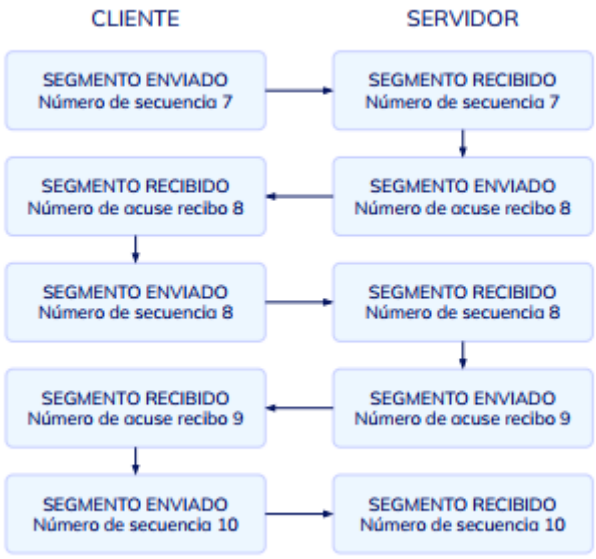
El **control de transferencia** se logra con estos dos campos.

4	32	Número de secuencia
8	64	Número de acuse de recibo (si ACK es establecido)

El host enviará el siguiente segmento siempre

que reciba un segmento de respuesta indicando el número del siguiente segmento que se debe enviar.

De esta forma el emisor no enviará el siguiente segmento hasta que el receptor le indique que ya ha recibido el anterior y está listo para recibir el siguiente.



Existen muchos motivos por los que el emisor **no** recibe el acuse recibo:

Exceso de tráfico en la red.

El firewall del destino está bloqueando un puerto o protocolo de transporte.

Los saltos de red exceden los tiempos de respuesta.

Otros.

En los casos que no haya respuesta, el emisor seguirá enviando el mismo segmento hasta que pasen 2 cosas:

Se reciba el acuse recibo y retome el envío de segmentos.

Se aborte la conexión pasado un tiempo de espera o Retransmission Timeout (RTO). La duración de la cuenta regresiva se adapta de forma dinámica mediante un algoritmo y depende de la velocidad individual de transmisión.

Orientado a conexión

TCP es un protocolo orientado a la conexión que aborda numerosos problemas de confiabilidad al proporcionar un flujo de bytes confiable.

Entre el host que envía y el host que recibe hay una conexión **estable** y **confiable** donde se envían mutuamente segmentos: el host de origen que envía segmentos que acarrean el mensaje y el host destino devuelve segmentos confirmando la recepción, explicándolo de forma resumida.

Básicamente bajo este protocolo se genera un canal de comunicación de puerto a puerto donde se intercambian datos, pero para lograr esto existen unos pasos previos en donde ambos hosts se reconocen mutuamente, se habilitan mutuamente para proceder al intercambio de datos y finalizan dicha conexión terminado el intercambio de datos y con ello la conexión.

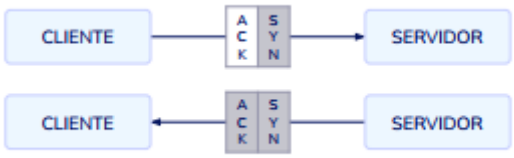
Para esto el segmento tcp tiene un conjunto de campos de 1 bit cada uno, también denominados 'flags', que determinan cada estado en el proceso de conexión.

Saludo de 3 vías

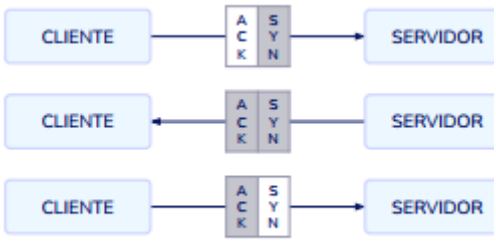
El protocolo TCP es orientado a conexión y para lograr esa conexión antes de enviarse los segmentos que contienen el mensaje debe establecerse la conexión, y esto se logra mediante un mecanismo llamado "**Saludo de 3 vías**" o "***3 way handshake***".

1. **Solicitud de conexión**: el emisor envía un segmento con el campo o *flag* '**SYN**' activado.

2. **Aceptación de conexión**: el receptor recibe la solicitud y la aprueba devolviendo un segmento con el *flag* '**ACK**' activado, y simultáneamente solicita conexión al emisor con el *flag* '**SYN**' activado.



3. El emisor recibe la confirmación de conexión aprobada por el receptor y responde la solicitud de conexión del receptor con un segmento con el *flag* '**ACK**' activado.



4. **Conexión establecida**: en este punto ambos hosts proceden al intercambio de datos. Cuando el intercambio finaliza se procede a la desconexión, un proceso similar al '**SYN-ACK/SYN-ACK**' pero con el *flag* '**FIN**'.

Protocolo de datagramas de usuario UDP

El **Protocolo de datagramas de usuario (UDP)** es un protocolo de datagramas **no orientado a conexión**. Al igual que IP, es un protocolo poco confiable. La confiabilidad se aborda mediante la detección de errores mediante un algoritmo de *checksum*. UDP se usa generalmente para aplicaciones como transmisión de medios (audio, video, voz sobre IP, etc.) donde la **llegada a tiempo** es más importante que la confiabilidad, o para aplicaciones simples de consulta / respuesta como búsquedas de DNS.

El **Protocolo de transporte en tiempo real (RTP)** es un protocolo de datagramas que se utiliza sobre UDP y está diseñado para datos en tiempo real, como medios de transmisión.

El protocolo UDP no utiliza mecanismos de conexión como el saludo de 3 vías. El objetivo es el de **mantener el tiempo de transmisión lo más bajo posible**.

El protocolo UDP funciona sin conexión: se caracteriza porque permite el envío de datagramas a través de la red sin que se haya establecido previamente una conexión entre el emisor y el receptor. Los datagramas respectivos se envían a la dirección IP preferida de la secuencia especificando el puerto de destino, sin que sea necesario que el ordenador asociado al mismo tenga que dar una respuesta.

No obstante, cuando los paquetes tienen que ser devueltos al emisor, existe la posibilidad de incluir en la cabecera UDP información sobre el puerto de origen.

Características

UDP utiliza puertos: al igual que el TCP, el protocolo UDP utiliza puertos para permitir que los datagramas se transfieran a los protocolos correctos, es decir, a las aplicaciones elegidas del sistema de destino. Los puertos quedan definidos mediante un número conforme a un rango de valores válidos, estando reservado el rango de 0 a 1023 para los servicios fijos.

El protocolo UDP permite una **comunicación rápida** y **sin retardos**: el protocolo de transporte es el adecuado para una transmisión de datos rápida debido a que no hay que llevar a cabo una configuración de la conexión. Esto resulta también del hecho de que la pérdida de un paquete individual afecta exclusivamente a la calidad de la transmisión. En el caso de conexiones TCP, en cambio, se intenta reenviar de nuevo los paquetes perdidos de forma automática, lo que provoca que todo el proceso de transmisión se detenga.

El protocolo UDP **no ofrece** ninguna garantía de **seguridad** e **integridad de los datos**: la ausencia de acuse de recibo mutuo entre el emisor y el receptor garantiza que la velocidad de transmisión en el protocolo UDP sea excelente; no obstante, el protocolo no puede garantizar la seguridad ni la integridad de los datagramas. Tampoco puede garantizar el orden de los paquetes enviados. Por ello, los servicios que utilizan UDP deben aplicar sus propias medidas de corrección y protección.

Aplicaciones del protocolo UDP

Aplicaciones basadas en ***best effort delivery***: el escenario típico en el que encontramos el protocolo UDP son las aplicaciones basadas en la **entrega de mejor esfuerzo**. A este tipo de programas, que utilizan el protocolo de datagramas de usuario como un mecanismo de mejor esfuerzo, les basta con una transmisión confiable de la información, porque la repiten constantemente de igual manera. Encontramos ejemplos en aquellas aplicaciones que transmiten valores medidos o que ejecutan repetidamente las mismas órdenes de trabajo.

Aplicaciones **ligeras**: la baja sobrecarga del protocolo de transporte proporciona un soporte óptimo para las aplicaciones que tengan un diseño muy sencillo. Lo anterior, junto con el hecho de que no es necesario configurar una conexión, hacen que estas aplicaciones puedan beneficiarse de un rendimiento especialmente alto en el procesamiento y reenvío de datagramas en las redes.

Aplicaciones con **mecanismos propios** para una **transmisión fiable**: el UDP puede resultar interesante para aquellas aplicaciones que necesitan un intercambio de información fiable, pero que dependen de sus propios mecanismos a la hora de dar respuesta a los datagramas. La ventaja de este tipo de servicios es que no están sujetos a patrones fijos a la hora de garantizar la integridad y exactitud de los datagramas enviados. Pueden decidir por sí mismos cómo y cuándo reaccionar ante información incorrecta o no clasificada.

Aplicaciones **multicast**: mientras que los protocolos de transporte confiables como el protocolo TCP se limitan a la comunicación de extremo a extremo, el protocolo UDP también soporta conexiones de **multidifusión IP**. En el caso de que una aplicación deba enviar paquetes IP de forma eficiente y rápida a varios receptores al mismo tiempo, el UDP proporciona el soporte adecuado.

Aplicaciones en **tiempo real** (*real time applications*): por último, el UDP también es adecuado como protocolo de transporte para servicios que presentan requisitos de **comunicación en tiempo real**, como las transmisiones de audio o vídeo. Se trata de protocolos que deben ser capaces de controlar en gran medida la transmisión, recepción y reproducción de flujos de datos por sí mismos, lo que resulta sencillo en las transmisiones basadas en el protocolo UDP sin conexión.

Capa 2: Internetworking o Internet

El **Protocolo de Internet** es el componente principal de la capa de Internet y define dos sistemas de direccionamiento para identificar los hosts de la red y ubicarlos en la red:

El sistema de direcciones original de ARPANET y su sucesor, Internet, es el **Protocolo de Internet versión 4 (IPv4)** la cual utiliza una dirección IP de **32 bits** y, por lo tanto, es capaz de identificar aproximadamente cuatro mil millones de hosts.

Esta limitación fue eliminada en 1998 por la estandarización del **Protocolo de Internet versión 6 (IPv6)** que usa direcciones de **128 bits**. Las implementaciones de producción de IPv6 surgieron aproximadamente en 2006.

En esta capa es donde los segmentos generados en la capa de transferencia son direccionados hacia los hosts correspondientes, tanto dentro de la red como fuera. A cada segmento se le imprime una dirección IP de origen y otra de destino, y así como los segmentos poseen una cabecera con puerto de origen y puerto destino en el caso de la capa de internet los segmentos son encapsulados en lo que se denomina "**datagrama**".

Segmentos encapsulados en paquetes de datos.

Paquete de datos: datagrama y cabecera IP

Paquete de red o paquete de datos es cada uno de los bloques en que se divide la información para enviar, en el nivel de red. El datagrama IP es la unidad básica de transferencia en una red IP. El datagrama consiste de una cabecera IP y de un campo de datos.

0-3	4-7	8-15	16-18	19-31
Versión	Tamaño cabecera	Tipo de servicio	Longitud final	
Identificador			Flags	Posición de Fragmento
Tiempo de vida		Protocolo	Suma de Control de Cabecera	
Dirección IP de Origen				
Dirección IP de Destino				
Opciones				Relleno

Cabecera de un datagrama IP.

Un paquete está generalmente compuesto de 3 elementos:

Una **cabecera** que contiene generalmente la información necesaria para trasladar el paquete desde el emisor hasta el receptor.

El **área de datos** (*payload*) que contiene los datos que se desean trasladar.

La **cola** (*trailer*), que comúnmente incluye código de detección de errores.

Direccionamiento: dirección de origen y destino

Quizás los aspectos más complejos de IP son el direccionamiento y el enrutamiento.

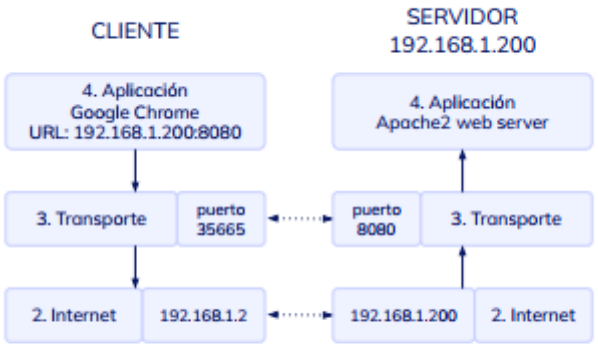
El **direccionamiento** se refiere a la forma como se asigna una dirección IP y cómo se dividen y se agrupan subredes de equipos. Básicamente, es asignar una dirección de origen y destino.

Para comprender mejor el proceso de envío y recepción de los mensajes podemos hacer la siguiente analogía:

Una persona escribe una carta. **Capa de aplicación**.

La carta se divide en varias hojas y se organiza en un orden correlativo. **Capa de transporte**.

La carta se guarda en un sobre (paquete de datos) y se indica en el sobre el remitente (dirección origen) y destinatario (dirección de destino). **Capa de internet**.



El paquete de datos sale de un host identificado por una IP y se entrega al host indicado en el destino del paquete.

Enrutamiento

En una red de datos existen varios mecanismos para que la entrega se realice, y eso depende de si el host destino se encuentra o no dentro del segmento de red.

Como se explicó anteriormente, la dirección IP es una dirección lógica que se le asigna a un host (id de red) dentro de una red lógica (determinada por la máscara de red). Un paquete no sabe cómo ir del punto A al punto C, solo contiene la información desde dónde parte y dónde quiere llegar.

Para el transporte de estos paquetes **hacen falta protocolos de enrutamiento y mecanismos físicos,** es por eso que esta capa está muy relacionada con la capa de red (1).

El paquete, por otra parte, no depende de las otras capas, **un paquete de datos no está determinado por lo que transporte.**

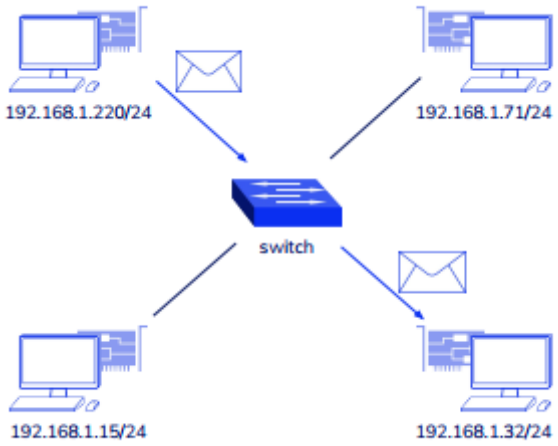
La entrega de paquetes se logra de varias formas y depende del destino, si está dentro o fuera del segmento lógico y el tipo de topología física.

Enviando paquete en la misma red lógica

Los paquetes se vuelcan a la red por un dispositivo de red apropiado (NIC) y los mecanismos físicos están determinados por el tipo de red y la tecnología de interconexión de nodos.

En una red Lan cuyos hosts están interconectados por un switch, los paquetes se entregan directamente al host destino. Tanto las NICs como los switches guardan información que relaciona la dirección IP y MAC de cada NIC de la red, de esta forma el conmutador cierra circuitos de host a host.

En el ejemplo vemos un paquete enviado del host 192.168.1.220 al host 192.168.1.32.



La tecnología usada anteriormente no tenía este nivel de inteligencia, el hub o concentrador recibía la información y la reenviaba a todos los hosts conectados, sencillamente los hosts descartan los paquetes cuya dirección IP de destino no fuese la propia.

Vemos como los paquetes cuya dirección de destino no coincide con la IP del host se descartan.

Enrutado de paquetes entre redes

Uno de los beneficios del protocolo IP es que permite su **enrutamiento**, es decir, lograr caminos que transporten los paquetes de una red lógica a otra.

El enrutamiento consiste en encontrar un camino que conecte una red con otra y, aunque es llevado a cabo por todos los equipos, es realizado principalmente por routers, que no son más que computadoras especializadas en recibir y enviar paquetes por diferentes interfaces de red, así como proporcionar opciones de seguridad, redundancia de caminos y eficiencia en la utilización de los recursos.

Cuando un paquete de datos tiene una dirección de destino que no pertenece a la propia red lógica del host emisor este paquete debe encaminarse, aquí entra el concepto de ***gateway***.

Gateway

Es el dispositivo de red que puede traducir paquetes que provienen de una red hacia otra red. El dispositivo usado es un router que tiene varias interfaces de red asociadas a redes distintas, el router sabe que camino elegir para que un paquete de datos llegue de una red a otra.

Para que esto suceda cada host debe tener configurado un *gateway* que es la dirección IP del dispositivo que le permitirá sacar paquetes de la red.

Un *gateway* modifica el empaquetamiento de la información de la red de origen para acomodarse a la estructura de la red de destino. El router recibe un paquete por una interfaz, lo traduce y lo envía por la interfaz adecuada.

Los routers se comunican entre sí informándose sobre otras redes y caminos a través de los protocolos de enrutamiento:

RIP: protocolo de portal interior (IGP) que enruta paquetes IPv4 y mantiene una tabla

de enrutamiento.

RDISC: les permite a los hosts descubrir la presencia de un router en la red.

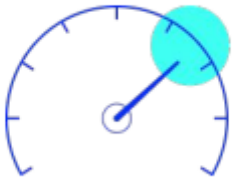
RIPng: IGP que enruta paquetes IPv6 y mantiene una tabla de enrutamiento.

ND: advierte la presencia de un router IPv6 y descubre la presencia de hosts IPv6 en una red.

Asociado al encaminamiento existe el concepto de **métrica**, que es una medida de lo "**bueno**" que es usar un camino determinado.

La **métrica** puede estar asociada a distintas magnitudes: distancia, coste, retardo de transmisión, número de saltos, etc., o incluso a una combinación de varias magnitudes. Si la métrica es el retardo, es mejor un camino cuyo retardo total sea menor que el de otro. Lo ideal en una red es conseguir el encaminamiento óptimo: tener caminos de distancia (o coste, o retardo, o la magnitud que sea, según la métrica) mínimos.

Típicamente el **encaminamiento** es una función implementada en la capa 3 (capa de red) del modelo de referencia OSI.



Capa 1: Acceso a la red

La **capa de acceso a la red** provee todos los aspectos que un paquete IP requiere para **efectuar un enlace físico**, un canal de transmisión de NIC a NIC.

Las funciones de la capa de acceso de red incluyen la asignación de direcciones IP a las direcciones físicas y el encapsulamiento de los paquetes IP en tramas.

En esta capa existen **dos sub capas**, la de Control de enlace lógico y la capa MAC.

Dirección física: MAC

Antes que nada debemos establecer que es una dirección **MAC**.

En las redes de computadoras, la **dirección MAC** (siglas en inglés de *Media Access Control*) es un identificador de 48 bits (6 bloques de dos caracteres hexadecimales [8 bits]) que corresponde de forma única a una tarjeta o dispositivo de red. Se la conoce también como **dirección física**, y es **única** para cada dispositivo. Está determinada y configurada por el IEEE (los últimos 24 bits) y el fabricante (primeros 24 bits) utilizando el *organizationally unique identifier*.

Una dirección MAC tiene el siguiente formato:

30:9c:23:a1:2d:ce

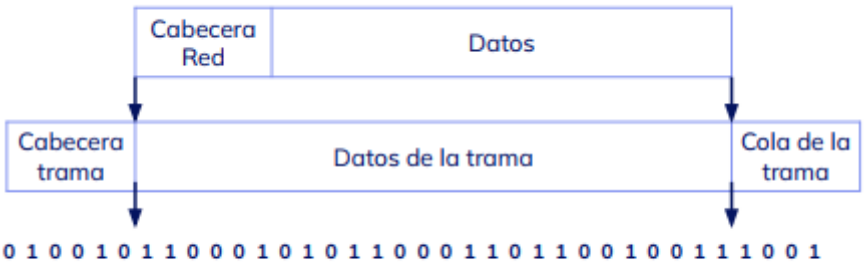
Está conformada por 6 números hexadecimales, y como se mencionó, cada interfaz tiene una que viene incorporada de fábrica.

A un host se lo identifica mediante una dirección IP, lo que en realidad, se le asigna a su NIC una dirección IP. Un host puede tener tantas direcciones IP como interfaces de red. Esta es la forma de identificar al host en una red lógica para que los paquetes puedan entregarse incluso a redes distintas, pero desde el punto de vista de transporte físico sucede diferente.

Tramas de red

Los mensajes fueron pasando desde su generación en la capa de aplicación, segmentación en la capa de transporte y el empaquetado en la capa de internet.

Finalmente ese paquete debe volcarse a un medio físico, dicho paquete ahora se encapsula en lo que se denomina **trama**.



Paquete siendo transportado en una trama.

Las tramas, al igual que otras unidades de datos, tienen **campos con información** como dirección MAC de origen y dirección MAC de destino, al ser direcciones físicas y no lógicas las tramas no pueden ir más allá del segmento de red físico.

En la capa de enlace los protocolos sólo actúan como máximo hasta la red local a la que está conectado un host cualquiera. Esto se denomina **enlace** si usamos el lenguaje propio de TCP/IP. Además esta capa se sitúa en la parte más baja de dicho modelo. Como dijimos esta capa tiene en cuenta todos los hosts accesibles en la red local o dicho de otra manera, todos los hosts que se pueden alcanzar sin tener que pasar por un enrutador.

Este modelo está diseñado para que el tipo de *hardware* usado no importe haciendo que pueda implementarse sobre cualquier tecnología de la capa de enlace. De hecho incluye también capas de los posibles enlaces virtuales que puedan haber ya sea por redes privadas virtuales y túneles de redes.

Funcionamiento

Subcapa LLC (Control de enlace lógico)

La **subcapa LLC** de Ethernet se ocupa de la **comunicación** entre las capas superiores y las capas inferiores. Generalmente, esta comunicación se produce entre el software de red y el hardware del dispositivo.

La subcapa LLC toma los datos del protocolo de la red, que generalmente son un paquete IPv4, y agrega información de control para ayudar a entregar el paquete al nodo de destino. El LLC se utiliza para comunicarse con las capas superiores de la aplicación y para la transición del paquete a las capas inferiores para su entrega.

El LLC se implementa en *software*, y su implementación no depende del *hardware*. En una PC, el LLC se puede considerar el controlador de la NIC. El controlador de la NIC es un programa que interactúa directamente con el hardware de la NIC para transmitir los datos entre la subcapa MAC y los medios físicos.

El control del enlace lógico:

Establece conexión con las capas superiores.

Encapsulan en tramas los paquetes de red.

Identifica el protocolo de la capa de red.

Subcapa MAC y acceso al medio

Esta capa además de identificar una trama con una dirección física de origen y destino provee un **mecanismo de acceso al medio** o ***Media Access Control*** **(MAC)**.

Si bien la tecnología ha cambiado, era un problema el hecho que varios hosts utilizaran el mismo medio físico al mismo tiempo en modelo de topología de bus o anillo, inclusive en redes interconectadas por HUBs: si dos o más hosts usaban el mismo medio en simultáneo las tramas colisionan y la información se pierde.

Actualmente se utilizan medios que permiten enviar y recibir información en **canales independientes**, es lo que se conoce como comunicación ***Full Duplex***.

Para solucionar este y otros problemas que presenta el direccionamiento físico se diseñó el protocolo MAC con las siguientes funciones:

Controlar el acceso al medio físico de transmisión por parte de los dispositivos que comparten el mismo canal de comunicación.

Agregar la dirección MAC del nodo fuente y del nodo destino en cada una de las tramas que se transmiten.

Al transmitir en origen debe delimitar las tramas de red agregando bits de bandera (*flags*) para que el receptor pueda reconocer el inicio y fin de cada trama.

Al recibir en destino debe determinar el inicio y el final de una trama de datos dentro de una cadena de bits recibidos por la capa física.

Efectuar detección y, si procede, corrección de errores de transmisión.

Descartar tramas duplicadas o erróneas.

Direccionamiento físico

Como último paso, los **datos** encapsulados en tramas, deben **volcarse a un medio**.

La interfaz de red es quien permitirá eso, convertir los bits en pulsos eléctricos en el caso de un medio de cobre. La subcapa mac se encargará de determinar el destino físico de la trama y cuándo y cómo utilizar el medio.

Protocolo ARP

Las interfaces de red guardan en su memoria una caché de direcciones MAC y la dirección IP asociada a dicha MAC de las NICs que pueda descubrir dentro de la LAN.

Básicamente tiene información acerca del resto de NICs conectadas a la red:

Dirección IP.

MAC asociada a esa dirección IP.

ARP se utiliza en cuatro casos referentes a la comunicación entre dos hosts:

● Cuando dos hosts están en la misma red y uno quiere enviar un paquete a otro.

● Cuando dos hosts están sobre redes diferentes y deben usar un gateway o router para alcanzar otro host.

● Cuando un router necesita enviar un paquete a un host a través de otro router.

● Cuando un router necesita enviar un paquete a un host de la misma red.

Dirección	TipoHW	DirecciónHW	Indic	Máscara	Interfaz
educacionit.example	ether	c0:8c:71:65:45:61	C		eno1
192.168.1.42	ether	f4:ec:38:cc:2d:5d	C		eno1
192.168.1.36	ether	04:b4:29:83:21:ef	C		eno1
_gateway	ether	e8:d1:1b:06:a3:ca	C		eno1
192.168.1.10	ether	54:e6:fc:b9:39:64	C		eno1
192.168.1.43	ether	30:9c:23:b2:b8:c1	C		eno1
192.168.1.37	ether	48:c7:96:31:71:3d	C		eno1

Caché con las direcciones MAC e IP de los hosts de una red LAN.

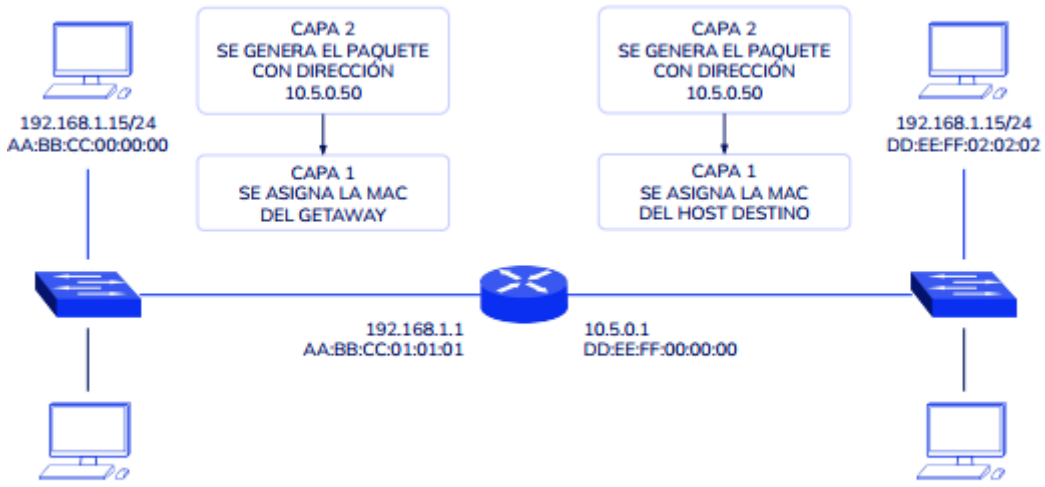
Direccionamiento de la trama

El proceso es el siguiente:

El paquete de datos se convierte en trama en la subcapa LLC.

La trama obtiene direcciones de origen y destino en la subcapa MAC, la subcapa superior informa sobre la dirección IP destino, la subcapa MAC se fija en su tabla ARP qué dirección física está asociada a dicha IP.

Se vuelca la trama al medio físico direccionada a la interfaz física correspondiente.



Una trama sólo puede llegar a destinos dentro del mismo segmento físico.

Modelo de capas OSI

Como se mencionó anteriormente, este es un modelo que **describe las etapas** en un proceso de comunicación, pero no termina el cómo. Un modelo de referencia nos ayuda a comprender los procesos y que se realiza en cada capa y nos proporciona los siguientes beneficios:

Facilita la comprensión al dividir un problema complejo en partes más simples.

Evita los problemas de compatibilidad, por ejemplo de red.

Detalla las capas para su mejor aprendizaje.

Proporciona a los fabricantes un **conjunto de estándares** que aseguran una mayor **compatibilidad** e **interoperabilidad** entre los distintos tipos de tecnología de red utilizados por las empresas a nivel mundial.

Capas del modelo OSI

El **modelo de interconexión de sistemas abiertos** (OSI) se divide en **7 capas** que podemos organizar en dos grupos:

Capas de host

Aplicación.

Presentación.

Sesión.

Transporte.

Capas de medio

Red.

Enlace de datos.

Física.

Cuando describimos los **elementos de una red** mencionamos los elementos que **hacen uso de una red** (tablets, pc, smartphones, etc), es decir los hosts, y los elementos que **proveen acceso a la red** y permiten el intercambio de información (NICs, SWITCHs, medios, etc.) entre hosts.

Es importante diferenciar entre estos dos niveles, ya que en ellos suceden procesos distintos.

A nivel de hosts se **generan** los mensajes, a nivel de medio se **transportan** esos mensajes entre los hosts.

Capa 7: Aplicación

En esta capa se ubican las **aplicaciones que hacen uso de los servicios de red**. Es donde el usuario u otros servicios generan los mensajes que pasarán a las capas inferiores.

Las aplicaciones de esta capa no solo generan los mensajes, sino que también los interpretan.

El ejemplo típico es el cliente de email: un usuario puede escribir y enviar un correo electrónico usando el cliente *Thunderbird*. El destinatario puede recibir y leer el correo electrónico usando *Outlook*.

Algunos ejemplos de aplicaciones que hacen uso de servicios de red:

Mensajería instantánea: Messenger, Telegram, Whatsapp, Signal.

Navegadores web: Firefox, Chrome, Internet Explorer, Edge, Opera.

Servidores: Mysql, Apache, Vsftpd, Nginx.

Videojuegos: League of Legends, Counter Strike, Minecraft.

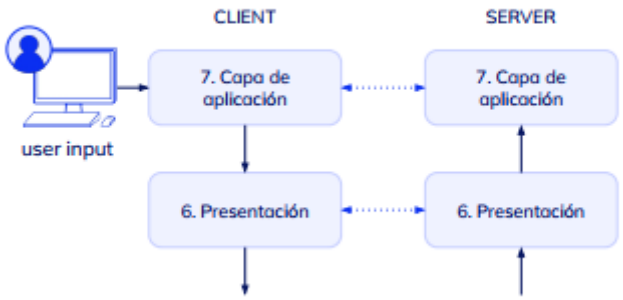
Streaming y videoconferencia: Zoom, Meets, Jitsi.

Vale decir que cuando hablamos de mensajes no nos referimos estrictamente a mensajes escritos, como un correo, sino a la **información** que genera la aplicación y que debe ser enviada a otro host.

Capa 6: Presentación

Su objetivo es encargarse de la **representación de la información**, de manera que aunque distintos equipos puedan tener diferentes representaciones internas de caracteres, números, sonido o imágenes, los datos lleguen de manera reconocible.

En esta capa es donde se comienzan a **aplicar** los **protocolos de aplicación**, continuemos con el ejemplo del correo electrónico y el de acceso a contenido web que utilizan protocolos abiertos.



Protocolo para transferencia simple de correo (SMTP)

Es el protocolo usado para el envío de correos electrónicos.

Redacción de mensaje

Mensaje escrito por el usuario mediante una aplicación de correo electrónico como Outlook.

Destinatario: jorge@example.com

Cuerpo del mensaje: “Hola ¿Qué tal tus vacaciones?”

Mensaje que se envía bajo el protocolo SMTP

La aplicación enviará un mensaje mucho más **complejo**, y normalmente, invisible al usuario.

Es un lenguaje que hablan entre aplicaciones, no entre usuarios.

```
S: 220 Servidor SMTP
C: HELO miequipo.midominio.com
S: 250 Hello, please to meet you
C: MAIL FROM: <yo@midominio.com>
S: 250 Ok
C: RCPT TO: <jorge@example.com>
S: 250 Ok
C: DATA
S: 354 End data with <CR><LF>.<CR><LF>
C: Subject: Campo de asunto
C: From: yo@midominio.com
C: To: jorge@example.com
C:
C: Hola ¿Qué tal tus vacaciones?
C: .
C: <CR><LF>.<CR><LF>
S: 250 Ok: queued as 12345
C: quit
S: 221 Bye
```

El Protocolo de transferencia de hipertexto (HTTP)

Mensaje enviado desde un navegador web

El mensaje que se envía es la url que se ingresa en la barra de direcciones de un navegador.

url: https://educacionit.com.ar

Mensaje que se envía bajo el protocolo HTTP

El mensaje que recibirá el servidor es la solicitud (GET) del contenido del directorio raíz (**/**) alojado en un servidor web bajo el nombre de host 'educacionit.com.ar'.

GET / HTTP/1.1

Host: educacionit.com.ar

Accept-Language: es

Capa 5: Sesión

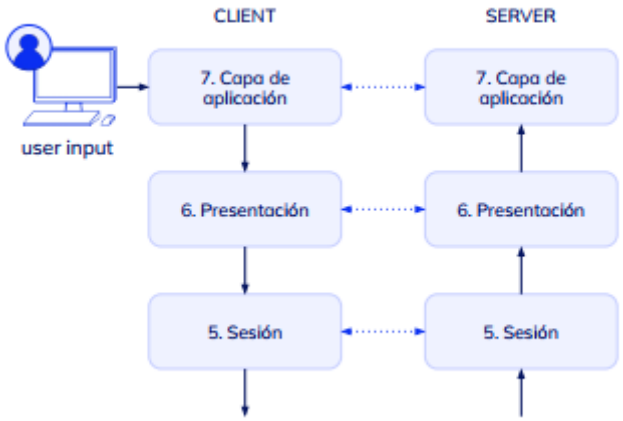
Establece, administra y finaliza las conexiones entre las aplicaciones locales y las remotas. Esta capa también permite:

Cifrar los datos y comprimirlos.

Establece y finaliza las conexiones.

Proporciona sus servicios a la capa de presentación.

Sincroniza el diálogo entre las capas de presentación de los dos hosts y administra su intercambio de datos.



En resumen, en esta capa se establecen los **mecanismos** que permiten que **las aplicaciones envíen y reciban los mensajes** entre ellas.

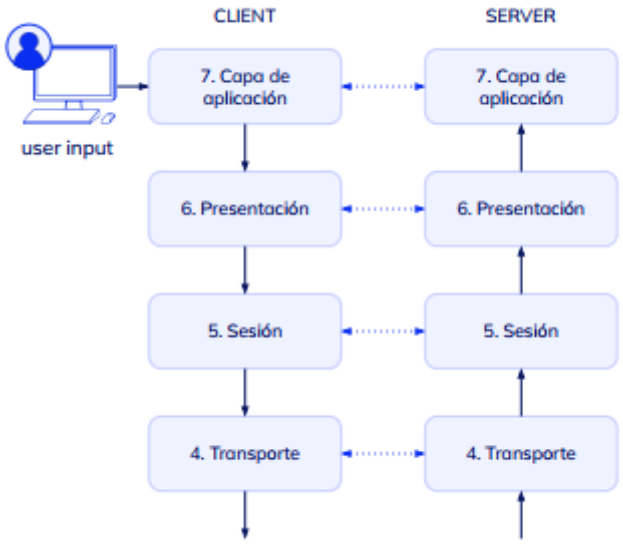
El tiempo que transcurre entre la apertura de la comunicación y el cierre de esta se conoce como **sesión**. La capa de sesión garantiza que la sesión permanezca abierta el tiempo suficiente como para transferir todos los datos que se están intercambiando.

Capa 4: Transporte

La capa de transporte es la responsable de las **comunicaciones de extremo a extremo** entre dos dispositivos.

El mensaje que viene de la capa de sesión no se envía como tal, sino que se fragmenta en lo que se denomina "**segmentos**", piezas más pequeñas que juntas componen el mensaje.

Los segmentos se reciben en el destino y el proceso de armado del mensaje a partir de los segmentos se produce en la misma capa en el extremo opuesto.



La capa de transporte es también la responsable del **control de flujo** y del **control de errores**.

El **control de flujo** sirve para determinar la velocidad óptima de transmisión que garantice que un emisor con velocidad de conexión alta no sobrecargue el segmento con menor capacidad de recepción, es decir, cuya conexión sea más lenta.

La **capa de transporte** realiza un control de errores en el extremo receptor, consistente en asegurarse de que todos los datos recibidos estén completos, y solicitará el reenvío en caso de que no.

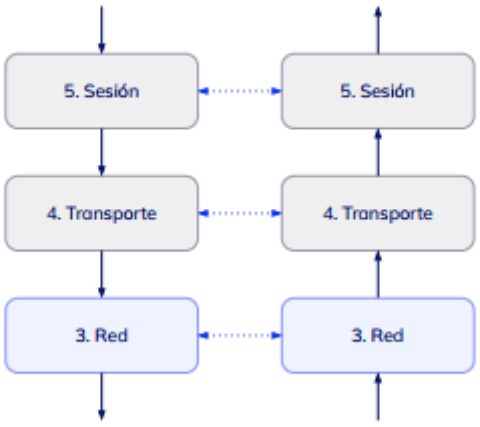
Capa 3: Red

En este punto entramos en el rango de capas de medio o "***media layers***".

Esta capa es responsable del **direccionamiento lógico** y el **dominio del enrutamiento**. Su misión es conseguir que los datos lleguen desde el origen al destino aunque no tengan conexión directa.

En esta capa los segmentos que llegan de la capa de transporte se encapsulan en lo que se denomina "paquete de datos", y llevan consigo dos datos fundamentales: la **dirección del host emisor** y la **dirección del host receptor**.

Esta información permitirá a los protocolos y dispositivos de enrutamiento hacer llegar los paquetes entre hosts que estén en redes distintas, o en el caso que se encuentren en el mismo segmento lógico, hacer llegar el paquete al host correspondiente dentro del mismo dominio de broadcast junto con los métodos de direccionamiento físicos descritos en capas inferiores.



Capa 2: Enlace de datos

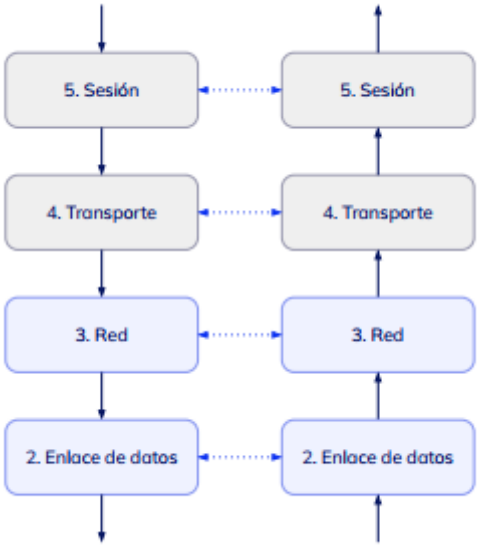
Proporciona **direccionamiento físico** y **procedimientos de acceso a medios**.

La capa de enlace de datos es muy similar a la capa de red, salvo que lo que hace es facilitar la transferencia de datos entre dos dispositivos ubicados en una misma red física.

La capa de enlace de datos toma los paquetes de la capa de red y genera trozos más pequeños denominados **tramas**. Estas tramas, así como los paquetes, tienen direcciones de origen y destino, pero en lugar de ser una dirección lógica es una dirección física, que es como se identifican las interfaces de red dentro de un segmento físico de red que se puede representar a partir de la topología.

El **direccionamiento lógico** permite a un paquete llegar entre segmentos de red físicos y lógicos distintos. Una trama solo llega a interfaces de red a las que se pueda acceder físicamente, que estén conectados entre sí, por ejemplo, los hosts que orbitan un switch.

Al igual que la capa de red, la capa de enlace de datos es también la responsable del control de flujo y de errores respecto de esa comunicación dentro de la red (la capa de transporte solo realiza esto último respecto de comunicaciones entre redes).



Capa 1: Física

Define todas las **especificaciones eléctricas y físicas de los dispositivos**.

Básicamente son los **medios de transmisión de datos**, por ejemplo:

Cables de cobre.

Cables de fibra óptica.

Ondas de radio (wifi, bluetooth, etc.)

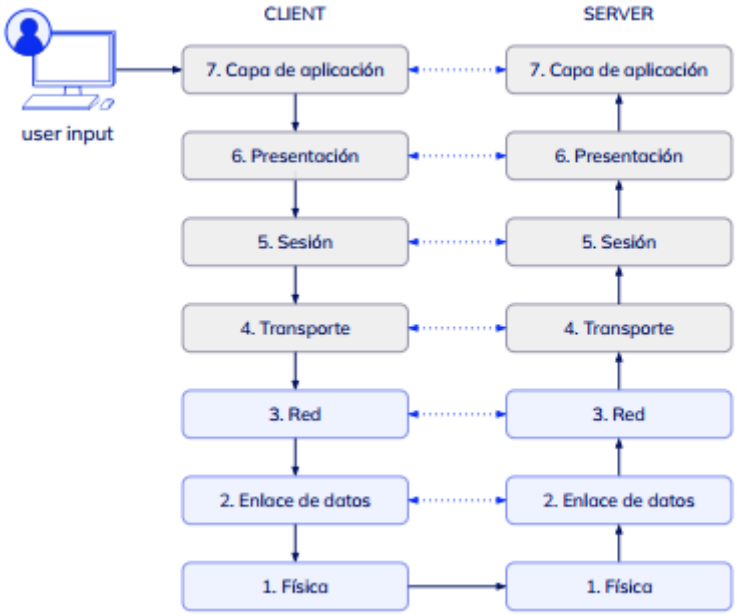
La capa física define las especificaciones eléctricas, mecánicas, de procedimiento y funcionales para activar, mantener y desactivar el enlace físico entre sistemas finales.

Las características tales como niveles de voltaje, temporización de cambios de voltaje, velocidad de datos físicos, distancias de transmisión máximas, conectores físicos y otros atributos similares son definidos por las especificaciones de la capa física.

Bluetooth es una especificación industrial para Redes Inalámbricas de Área Personal que posibilita la transmisión de voz y datos entre diferentes dispositivos.

ADSL Consiste en una transmisión analógica de datos digitales apoyado en el par simétrico de cobre que lleva la línea telefónica convencional.

USB es un estándar industrial desarrollado en los años 1990 que define los cables, conectores y protocolos usados en un bus para conectar, comunicar y proveer de alimentación eléctrica entre ordenadores, periféricos y dispositivos electrónicos. Consiste en una transmisión analógica de datos digitales apoyada en el par simétrico de cobre que lleva la línea telefónica convencional.



Conclusión

El **modelo OSI** describe el proceso de comunicación, no como se desarrolla ese proceso ni protocolos intervinientes ni tecnologías utilizadas.

De esta forma el modelo permite aislar el proceso de comunicación en capas que pueden ser observadas individualmente. Si una aplicación que debe enviar un mensaje a otra no lo está logrando, inspeccionar lo que se sucede en cada capa puede llevarnos a la identificación del problema y su resolución, desde la verificación del correcto conexionado e integridad de cables (capa 1) hasta las configuraciones de red de la aplicación (capa 4).

El modelo que se describe es **simétrico**, debe funcionar de igual manera en ambos sentidos y cada capa procesa la información generada en la misma capa del extremo opuesto.