Capítulo 1

Introducción a las Redes de Computadoras – Parte 2

Metas de la introducción

- **Agenda**:

1. Comprender los distintos **tipos de redes de computadora.**
2. Entender la arquitectura de los **sistemas operativos de redes (SOR).**
3. Aprender fundamentos sobre **cómputo en la nube**
4. Entender algunas convenciones a respetar en la materia

¿Por qué estudiar sistemas operativos de redes?

- Para manejar los tipos de redes que estudiamos hacen falta **sistemas operativos de redes (SOR).**
- En cada tipo de red hay **problemas** a ser resueltos si queremos que no tenga un mal desempeño y los SOR se encargan de resolver esos problemas.
  - Estudiarán cómo resolver problemas en los tipos de redes por medio de SOR.
- Para que las máquinas en un tipo de red se puedan comunicar hacen falta **protocolos de comunicación**; los SOR contienen esos protocolos.
  - Estudiarán cómo diseñar protocolos que permiten la comunicación entre hosts en los distintos tipos de redes.

Sistemas Operativos de Redes

- Los **sistemas operativos de redes (SOR)** están organizadas como una **pila de capas o niveles**, cada una construida arriba de la que está debajo de ella.
- La cantidad de capas, los nombres de las capas, sus contenidos y su función, difieren de un tipo de red a otro.

Sistemas Operativos de Redes

- **Metas**:
  1. Comprender la arquitectura de los SOR.
  2. Entender cómo funcionan las capas para el envío de mensajes.
  3. Comprender **problemas de diseño** a resolver en distintas capas.
  4. Estudiar los distintos **tipos de capas**
  5. Comprender el **modelo híbrido** de SOR que estudiaremos en la materia.

Jerarquías de Protocolos

- **Pilas de capas** se usan para reducir la complejidad del diseño de los SOR.
- ***¿Han estudiado arquitecturas de capas en alguna materia?¿Cómo era?***

Jerarquías de Protocolos

- **¿Cuál es el propósito de una capa en arquitecturas multicapa?**
  - ofrecer ciertos servicios a las capas superiores
  - ocultar la implementación a las capas superiores
- **Interfaces entre capas** = operaciones y servicios primitivos ofrecidos por una capa a capa superior.
- **Un SOR se preocupa de la comunicación de información.**
- **¿Cómo afecta esto al propósito de las capas?**
  - Una capa *n* se piensa como una conversación entre la capa *n* de una máquina con la capa *n* de otra máquina,
    - sin tener que preocuparnos de ciertos problemas que resuelven las capas inferiores a la capa *n*.
  - Para especificar cómo es esta conversación se definen **protocolos**

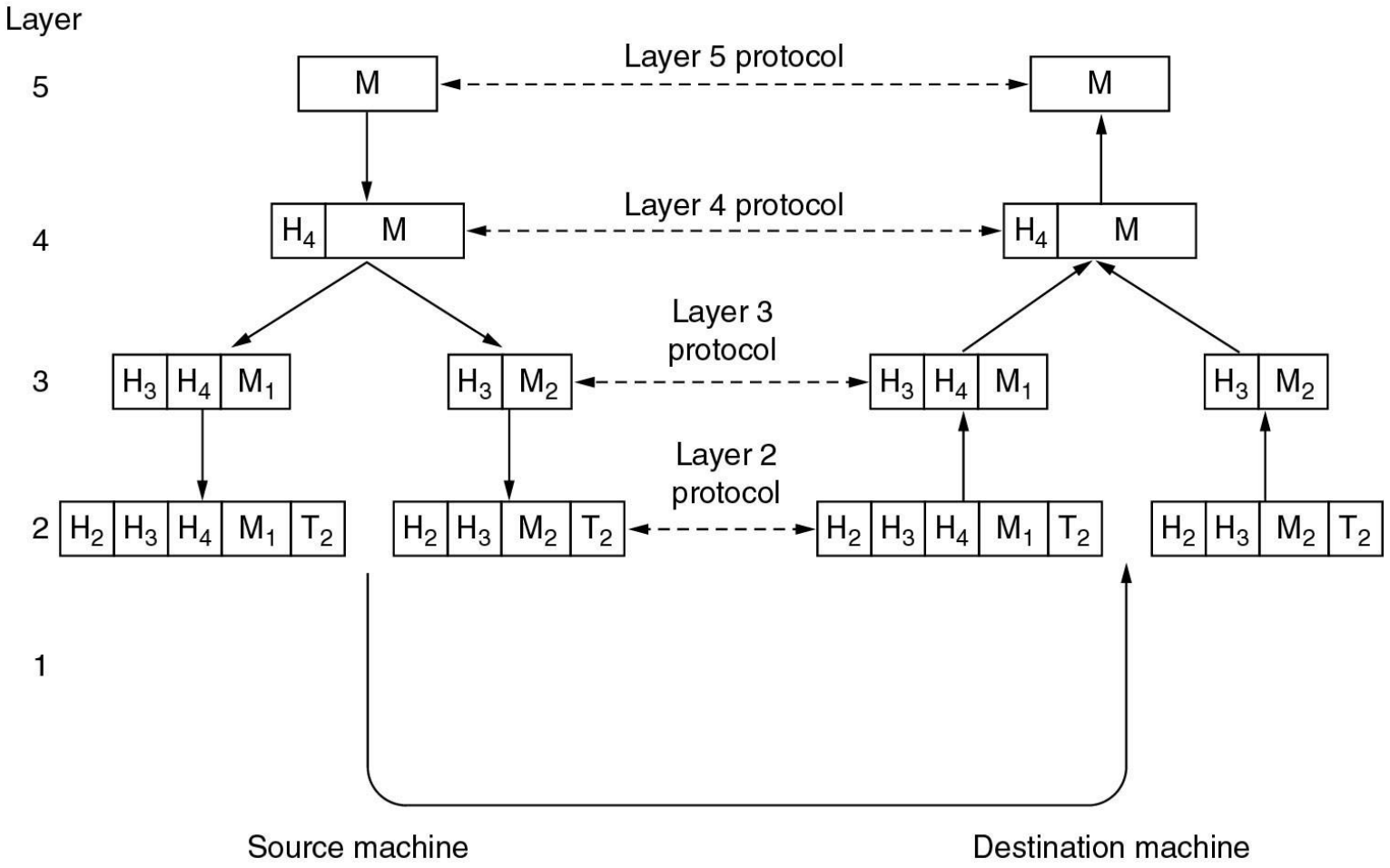
Jerarquías de Protocolos

- **Protocolo de capa *n*** = reglas y convenciones usadas en la **conversación** entre la capa *n* de una máquina y la capa *n* de otra máquina.
- **¿Cuándo ocurren comunicaciones entre capas consecutivas?**
  - **Durante el envío de mensaje**: cada capa pasa los datos y la información de control a la capa inmediatamente inferior, hasta que se alcanza la capa más baja.
  - **Durante la recepción de mensaje**: cada capa pasa cierta información conteniendo los datos a la capa inmediatamente superior hasta que alcanza la capa más alta.
- Debajo de la capa 1 está el **medio físico**.
- **arquitectura de red** = conjunto de capas y protocolos = **pila de protocolos**.

Sistemas Operativos de Redes

- **Metas**:
  1. Comprender la arquitectura de los SOR.
  2. Entender cómo funcionan las capas para el envío de mensajes.
  3. Comprender **problemas de diseño** a resolver en distintas capas.
  4. Estudiar los distintos **tipos de capas**.
  5. Comprender el SOR modelo TCP/IP
  6. Comprender el **modelo híbrido** de SOR que estudiaremos en la materia.

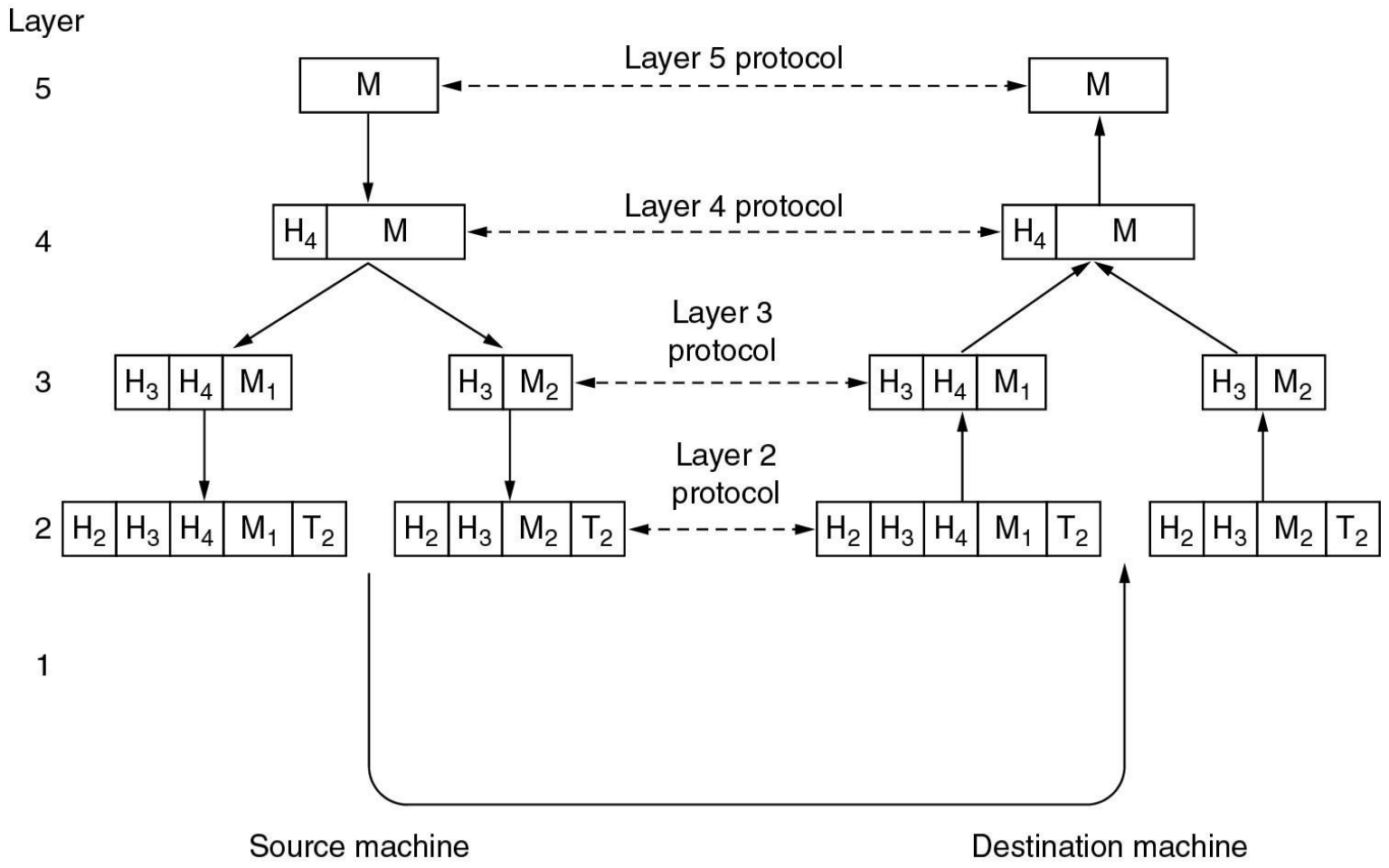
Jerarquías de Protocolos



Procesos de aplicación (capa 5 o **capa de aplicación**)

- P.ej. browser, e-mail, chat, ftp, etc.
- Produce un mensaje y lo pasa a la capa 4 para su transmisión.

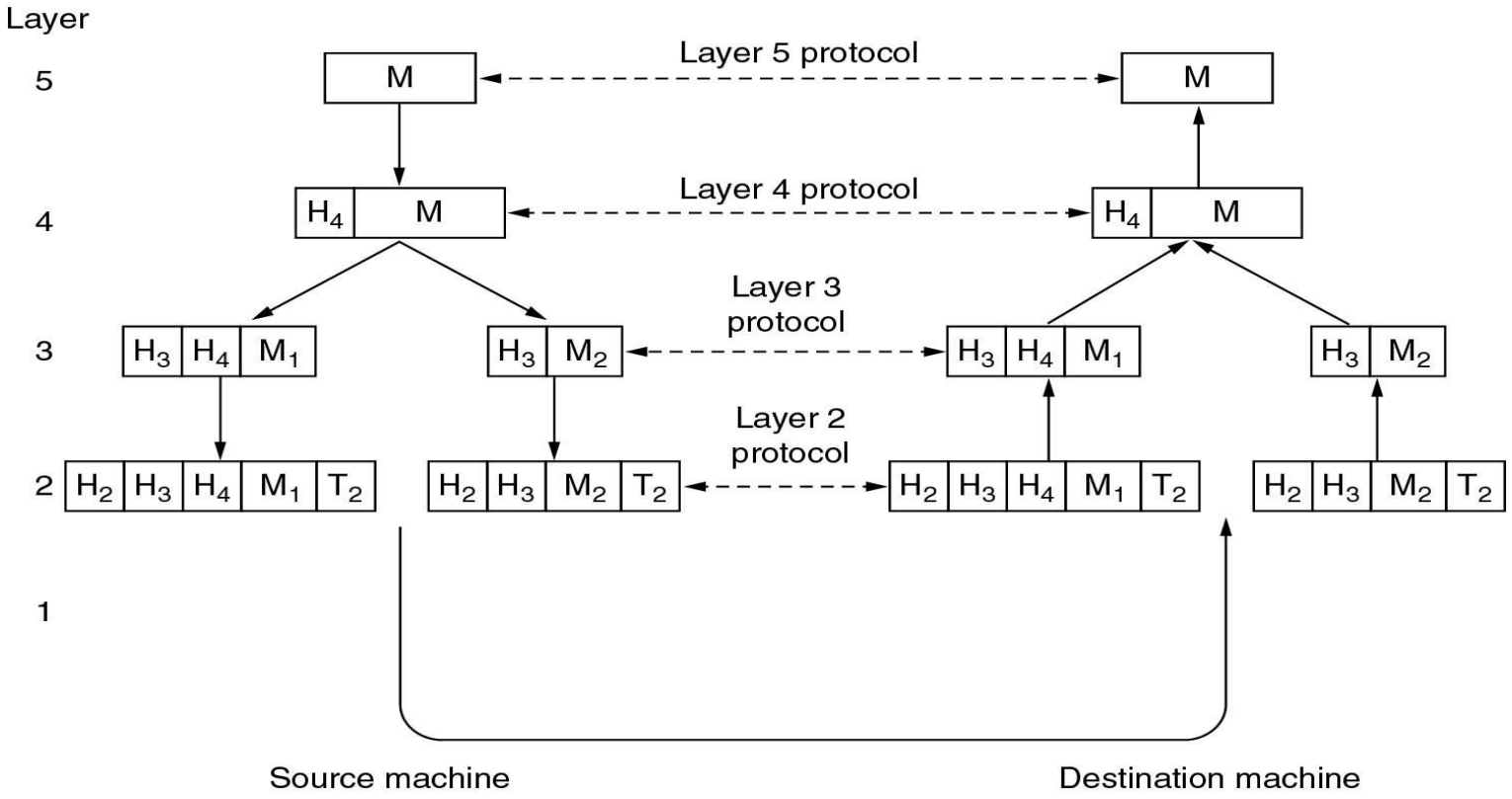
Jerarquías de Protocolos



La capa 4 (**capa de transporte**)

- pone un encabezado en el mensaje para identificarlo y pasa el resultado a la capa 3.
- El encabezado contiene **números de secuencia** para que la capa 4 en la máquina de destino entregue los mensajes en el orden correcto.

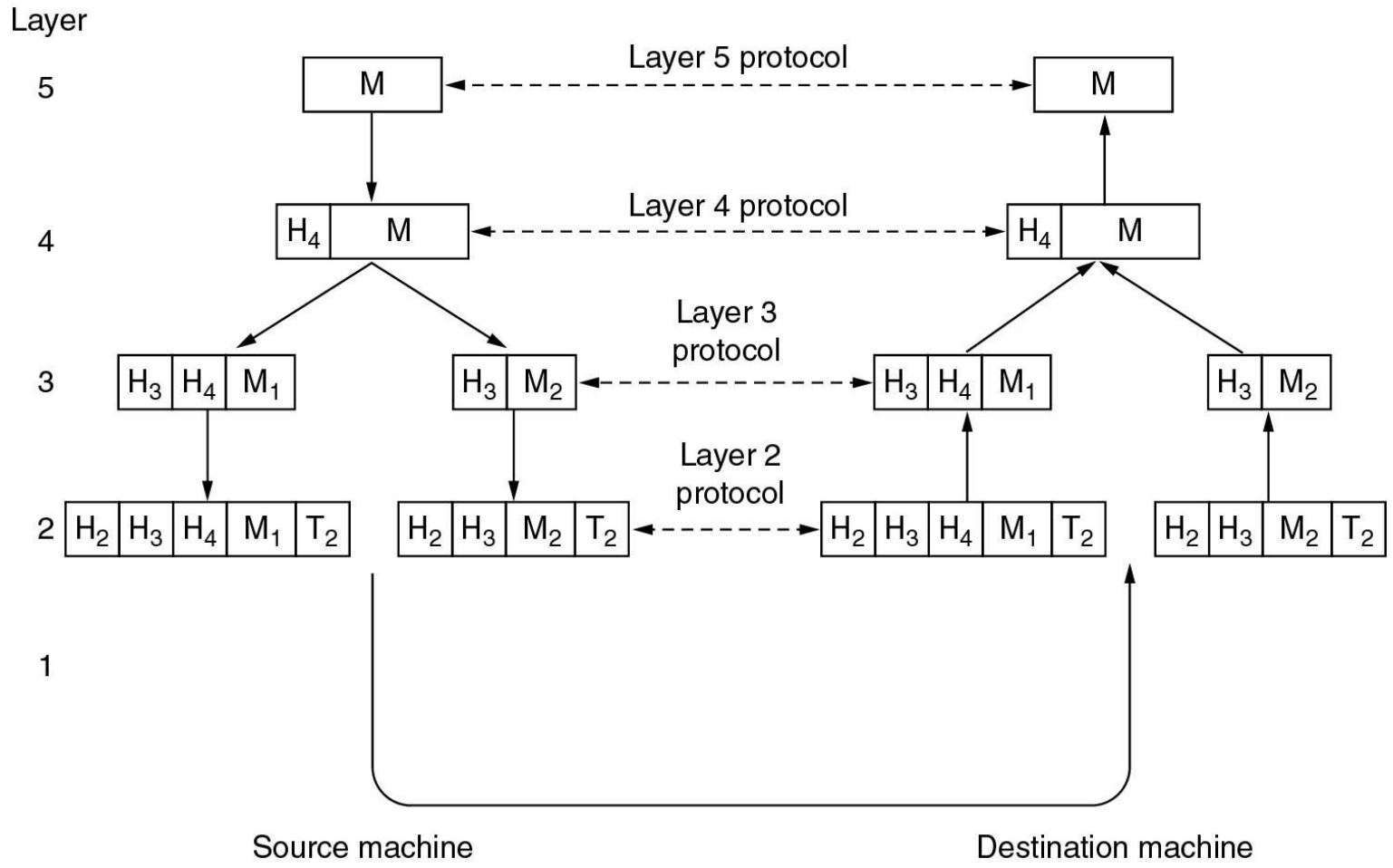
Jerarquías de Protocolos



Capa 3 (**capa de red**):

- Hay limitaciones en el tamaño de los mensajes de capa 3.
- Divide en **paquetes** los mensajes que llegan.
- A cada paquete se le coloca un encabezado.
- Decide cuál de las líneas que salen usar (cuando la máquina es un enrutador).
- Pasa los paquetes a la capa 2.

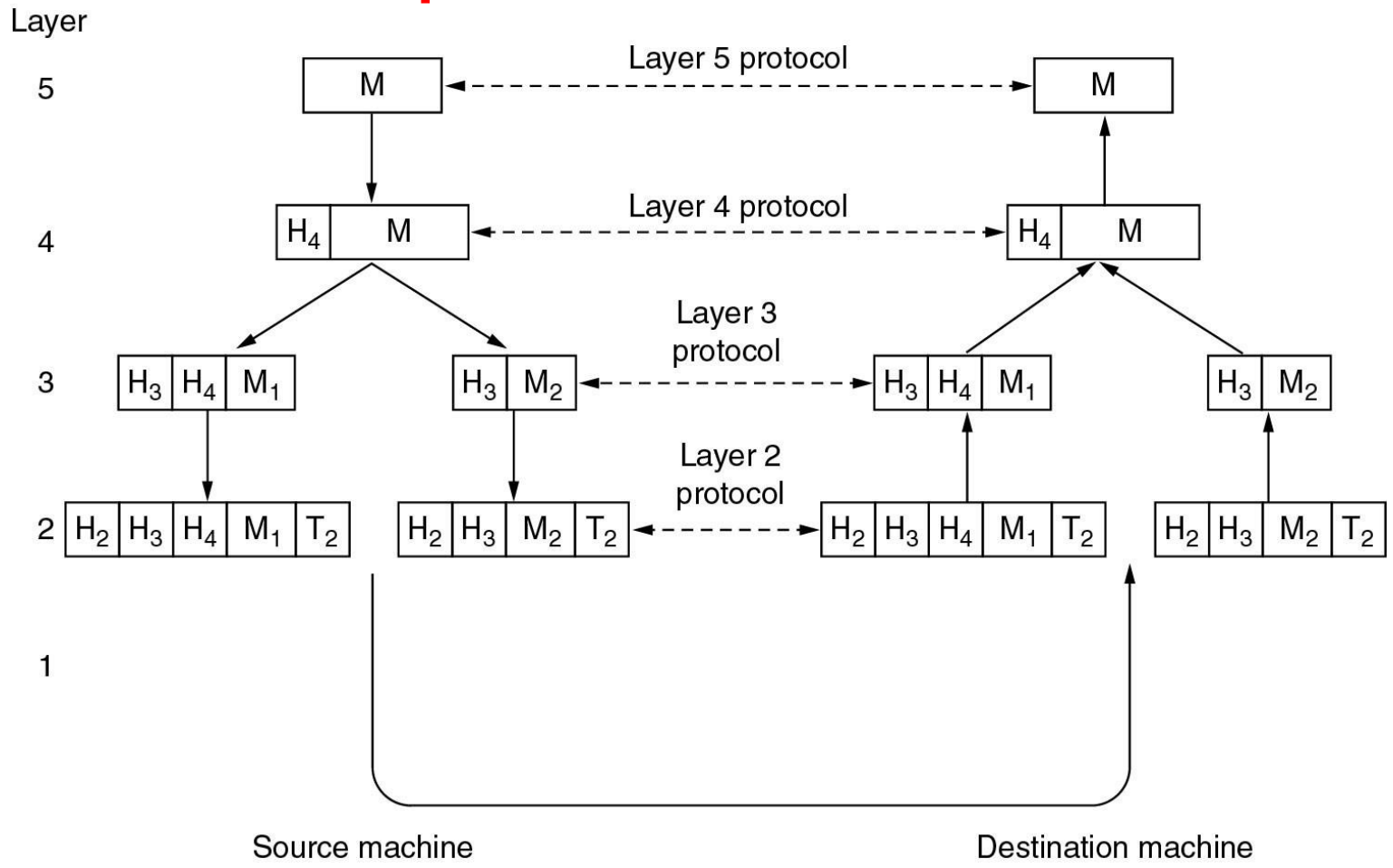
Jerarquías de Protocolos



La capa 2 (**capa de enlace de datos**)

- agrega un encabezado y un terminador, a cada pieza
- pasa la unidad resultante a la capa 1 para su transmisión.

Jerarquías de Protocolos



En la máquina receptora el mensaje pasa hacia arriba de capa en capa, perdiendo los encabezados conforme avanza.

Sistemas Operativos de Redes

- **Metas**:
  1. Comprender la arquitectura de los SOR.
  2. Entender cómo funcionan las capas para el envío de mensajes.
  3. Comprender **problemas de diseño** a resolver en distintas capas.
  4. Estudiar los distintos **tipos de capas**
  5. Comprender el **modelo híbrido** de SOR que estudiaremos en la materia.

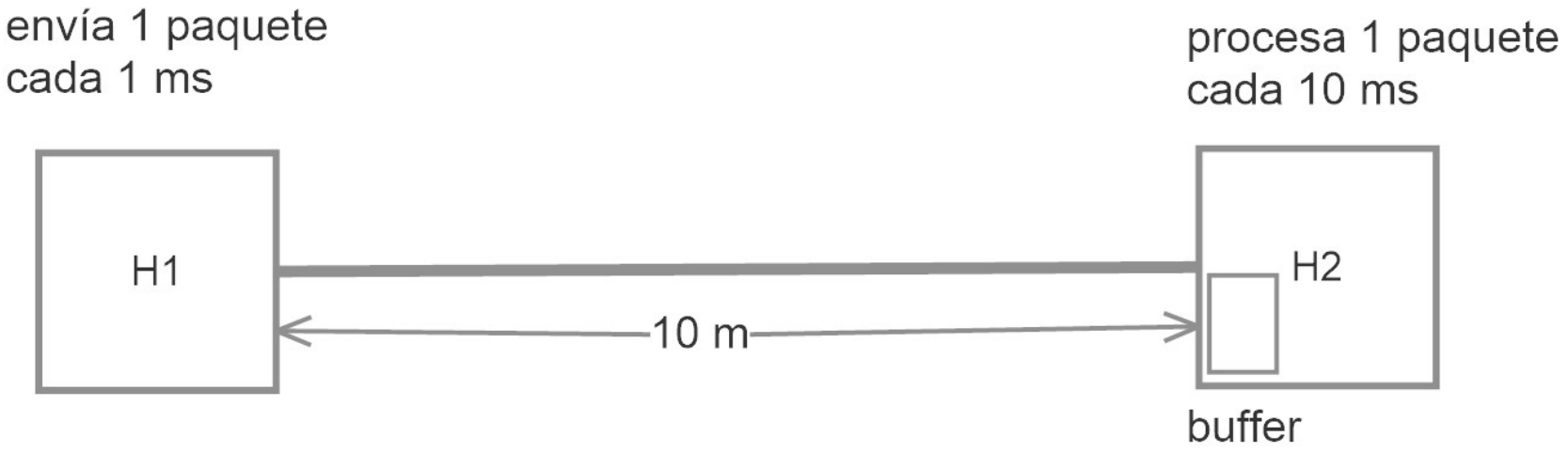
Aspectos de Diseño de las Capas

- Ahora vemos **problemas a resolver** por más de una capa.
- **Problema**: Hace falta un mecanismo para identificar a las máquinas de una red.
- **Solución**: Se usan direcciones para las máquinas.

Aspectos de diseño para las capas

- **Control de flujo**:
  - **Situación indeseable**: mensajes que llegan al receptor se pierden.
  - **¿Por qué se imaginan que puede pasar esto?**
  - **Causa**: un emisor rápido satura de datos al receptor hasta que este ya no puede almacenar más datos que le llegan y comienza a perder datos.

Aspectos de diseño para las capas



Aspectos de diseño para las capas

- **Control de flujo**:
  - **Problema: ¿Cómo evitar que un emisor rápido sature de datos a un receptor lento?**
  - **Idea de Solución**: Uso de **retroalimentación al emisor.**
    - O sea, indicarle cuándo y cuánto puede enviar.

Aspectos de diseño para las capas

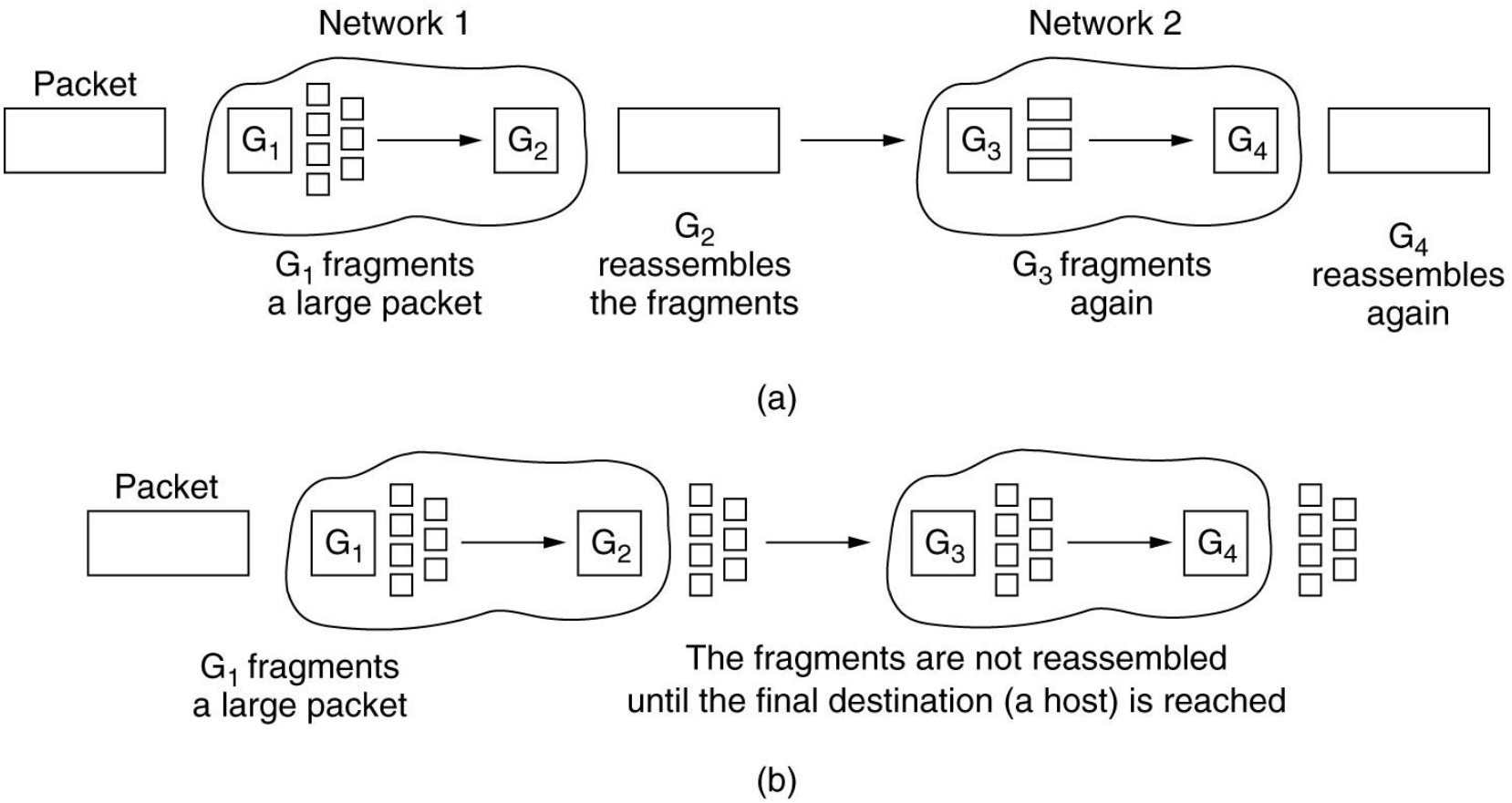
- **Fragmentación de mensajes**
  - Es común que las capas imponen un **tamaño máximo** a los mensajes. **¿Por qué?**
  - **Situación indeseable**: mensajes que llegan no pueden ser aceptados en una capa.
  - **¿Por qué puede pasar esto?**
  - **Causa**: los procesos son incapaces de aceptar mensajes que superan una cierta longitud
    - por ejemplo en la capa de enlace de datos y en la capa de red
    - Por ejemplo en una inter-red (≠ redes aceptan ≠ tamaños max)

Aspectos de diseño para las capas

- **Fragmentación de mensajes**
  - **Problema: ¿Cómo tratar un mensaje demasiado largo?**
  - **Idea de solución**: fragmentar mensajes, transmitir fragmentos y re-ensamblar mensajes.
  - **¿En una interred qué soluciones les llegan a la mente?**

Aspectos de diseño para las capas

Fragmentación de mensajes



(a) Fragmentación Transparente. (b) Fragmentación no transparente.

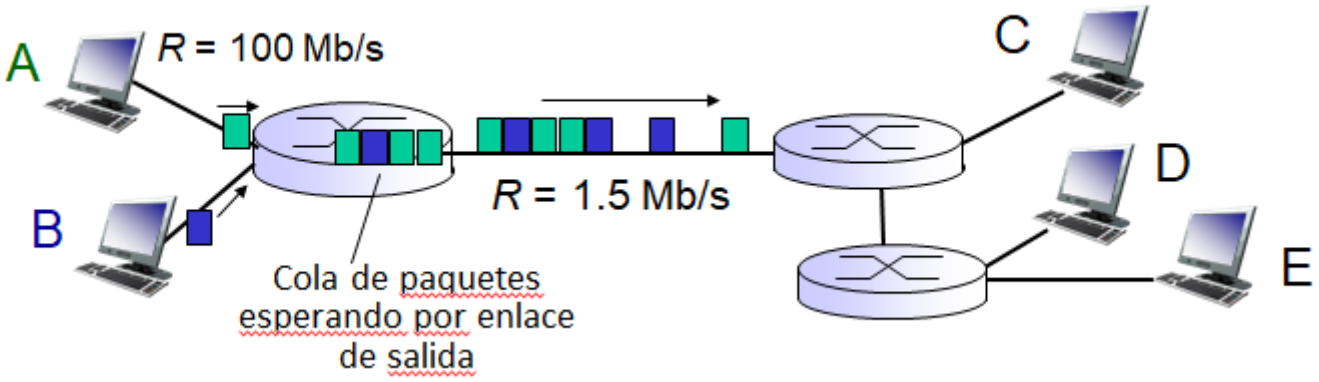
Aspectos de diseño para las capas

- **Congestión**
  - Debido a las limitaciones de los enrutadores y las líneas de salida una red tiene una determinada capacidad de conducción de mensajes..
  - **¿Qué limitaciones son esas?**
  - **Situación indeseable**: los mensajes enviados de host de origen a destino se pierden antes de llegar o demoran demasiado en llegar

Aspectos de diseño para las capas

Congestión por tráfico de un único nodo

Congestión por tráfico de múltiples nodos



Aspectos de diseño para las capas

- Los anteriores son ejemplos de **congestión**
  - La red no puede manejar la carga de paquetes que recibe de manera aceptable (esperas inaceptables o pérdida de paquetes)
- **Problema: ¿Cómo controlar la congestión?**
- **Idea de solución:** que máquinas emisoras se enteren de la congestión y reduzcan el tráfico de salida.

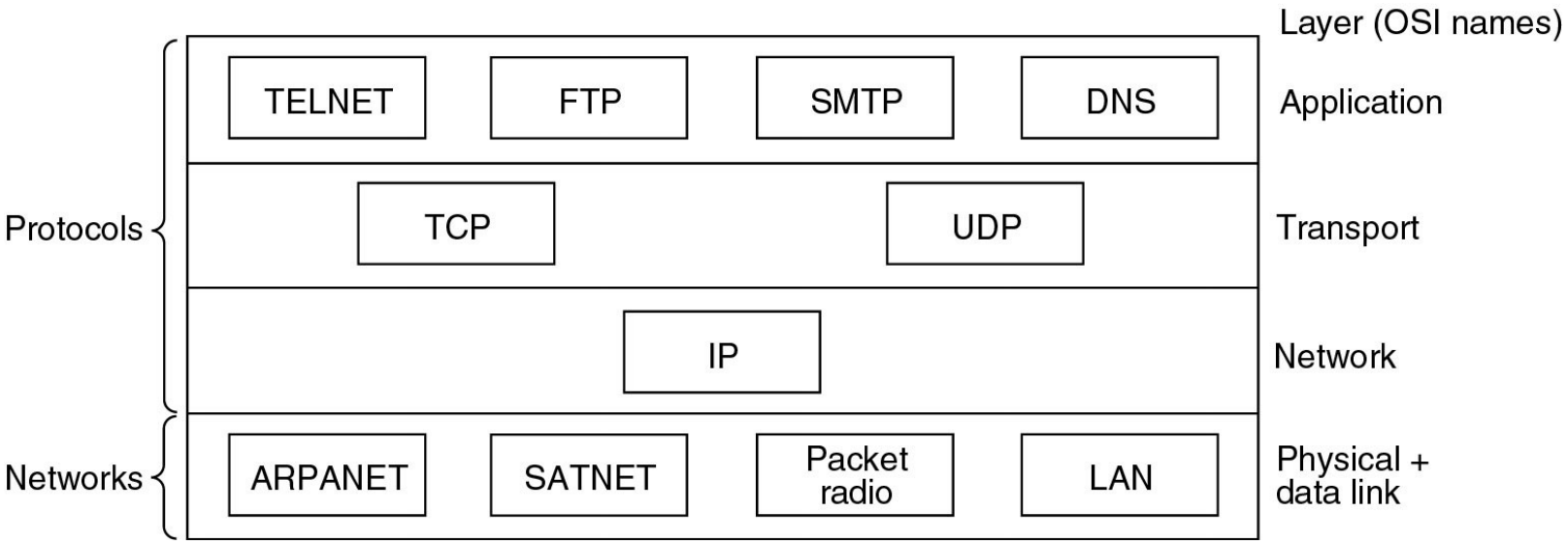
Sistemas Operativos de Redes

- **Metas**:
  1. Comprender la arquitectura de los SOR.
  2. Entender cómo funcionan las capas para el envío de mensajes.
  3. Comprender **problemas de diseño** a resolver en distintas capas.
  4. Estudiar los distintos **tipos de capas**
  5. Comprender el SOR modelo TCP/IP
  6. Comprender el **modelo híbrido** de SOR que estudiaremos en la materia.

Estudio de las capas

- **Plan**
  - Iremos estudiando las diferentes capas e
  - ilustraremos aquellas que lo permitan con el modelo de referencia TCP/IP.

Modelo de referencia TCP/IP



Capa de aplicación

- En la capa de aplicación tenemos las **aplicaciones de red.**
- Cada aplicación de red ofrece un servicio específico con su propia forma de interfaz con el usuario.

Capa de aplicación

- Hay **dos opciones para desarrollar aplicaciones de red**:

1. El programador para especificar la comunicación usa **una interfaz para programas de aplicación (API)**.
   - Una API es conjunto básico de funciones a ser usadas.
   - La **socket API** es el estándar de facto para el software que se comunica sobre la internet.
     - Será tema de estudio en el taller de la materia.
   - **¿Qué utilidad tiene saber sobre el SOR subyacente?**
   - El conocimiento del SOR subyacente permite al programador escribir mejor código y desarrollar aplicaciones más eficientes.

Capa de aplicación

2. El programador se apoya en **middlewares** para construir la aplicación de red.
   - **Por ejemplo**: la web, llamada a procedimientos/métodos remotos, etc.
   - Una **middleware** provee servicios al software de la aplicación que
     - hacen más fácil a los desarrolladores implementar la comunicación y la entrada/salida de modo que
     - se pueden enfocar en el propósito específico de la aplicación.

La capa de aplicación: TCP/IP

- La **capa de aplicación en TCP/IP** contiene varios protocolos de nivel mas alto: transferencia de archivos (FTP), correo electrónico (SMTP), para resolución de nombres de host en sus direcciones de red (DNS), para páginas web (HTTP), etc.

Capa de transporte

- La capa de red provee comunicación **entre hosts**
  - los paquetes de la comunicación entre los hosts siguen rutas elegidas por la capa de red.
- ***¿Con esto alcanza?***
- No, en la práctica la comunicación ocurre entre procesos.
  - La **capa de transporte** (**CT**) provee comunicación **entre procesos**.
- La CT **mejora** los servicios de la CR como veremos.
- La CT se ejecuta por completo en los hosts.
- **Entidad de transporte (ET)** = software/hardware de la CT.

Capa de transporte

- **¿Qué cosas se debería solucionar la CT?**
  - Uso de **temporizadores** y las **retransmisiones de paquetes**.
  - Uso de búferes y control de flujo.
  - Evitar congestionar la red poniendo demasiados paquetes en ella.
    - Cuando la CR pierde paquetes, la CT puede solucionarlo.
  - ***¿Si consideramos que hay procesos cliente y procesos servidor, qué otros problemas surgen?***
  - El direccionamiento explícito de los destinos.
    - ¿Cómo hacer para que un **proceso** adecuado **atienda a las necesidades** de una máquina **cliente**?
    - El proceso podría **no estar activo**, el cliente podría **no saber** cuál proceso usar, etc.

La capa de transporte: TCP/IP

- **Capa de transporte**. Tiene dos protocolos:
- **TCP**
  - TCP divide el flujo de bytes entrantes en mensajes discretos y pasa cada uno de ellos a la capa de interred.
  - TCP proporciona entrega confiable y en orden de los mensajes.
  - **¿Qué significa esto?**
  - Permite que un flujo de bytes que se origina en una máquina se entregue **sin errores** en otra máquina en la interred.
  - **Reensamblaje** de los mensajes recibidos en el receptor.
  - **¿Qué hacer para entrega confiable de mensajes?**
  - Mensajes recibidos son confirmados.
  - TCP también maneja el **control de flujo** y **el control de congestión.**

La capa de transporte: TCP/IP

- **UDP**
  - UDP proporciona entrega de mensajes no confiable y desordenada.
  - **¿Qué significa esto?**
  - Un mensaje puede entregarse con errores, o no entregarse, o varios mensajes pueden entregarse en forma desordenada.
  - **¿Entonces qué cosas de TCP puedo sacar de UDP?**
  - Mensajes recibidos no son confirmados.
  - Control de flujo, control de congestión, retransmisiones cuando se recibe mensaje erróneo.
  - **¿Entonces para que tipo de aplicaciones se puede usar UDP?**
    - Se usa para aplicaciones que no usan el control de flujo ni la secuenciación de mensajes.
    - Uso en consultas de solicitud-respuesta y en aplicaciones de transmisión de voz y video.

Capa de Red

- Objetivos de la **capa de red (capa 3)**.
  - Algoritmos de almacenamiento y reenvío
  - **Control de congestión.**
  - Resolver problemas que surgen cuando un mensaje tiene que viajar por redes de distinta tecnología para llegar a destino.

Capa de Red

- **Enrutamiento**
  - **Situación indeseable**: un mensaje demora demasiado en llegar
  - **¿Y esto por qué sucede?**
  - **Causa**: en determinadas redes (p.ej. WAN, internet, etc.) hay múltiples rutas entre el origen y el destino
    - Y justo se toma una ruta demasiado lenta/larga entre origen y destino
  - **¿Entonces cuál es el problema a resolver?**
  - **Problema**: ¿Cuando hay múltiples rutas entre el origen y el destino cómo elegir la mejor o las mejores?
    - De esto se encargan los **algoritmos de enrutamiento**

La capa de red: TCP/IP

- **Capa de interred**:
  - permite que los hosts inyecten paquetes dentro de cualquier red,
  - **¿Cómo viajan paquetes diferentes entre dos hosts?**
  - Estos viajan a su destino de manera independiente.
  - **¿Qué consecuencias tiene esto?**
  - Paquetes pueden llegar en un orden distinto al cual fueron enviados.
  - **¿Qué se hace con los paquetes que llegaron fuera de orden?**
  - las capas mas altas deberán ordenarlos, si se desea una entrega ordenada.

La capa de red: TCP/IP

- **¿Cómo se distingue entre diferentes máquinas (que tienen una conexión a internet)?**
- **Direcciones IP**
  - 4 números entre 0 y 255 separados por '.' .
  - P. ej. 200, 45.191.35
- **¿Cómo son los paquetes que se envían?**
- **Paquetes IP.** (tienen su propio formato como veremos más adelante.)
- **¿Cómo se hace el enrutamiento?**
- Hay protocolos de enrutamiento: se usan **OSPF** y **BGP** para enrutamiento de paquetes.

Procesos en comunicación

- ***procesos:*** programas ejecutándose dentro de un host
- Dentro del mismo host, dos procesos se comunican usando **comunicación inter-procesos** (definida por el sistema operativo)
- Los procesos en diferentes hosts se comunican intercambiando **mensajes**

clientes, servidores

Proceso cliente: proceso que inicia la communicación

Proceso servidor: proceso que espera ser contactado

Direccionando Procesos

- Para recibir mensajes, los procesos deben tener un *identificador*
- Un host tiene una dirección IP única de 32-bits
- **¿Es la dirección IP del host en el cual ejecuta un proceso suficiente para identificar el proceso?**
  - no, porque muchos procesos pueden estar ejecutándose en el mismo host.
- **¿Cómo se identifican los procesos?**
- *Identificadores de proceso* incluyen tanto direcciones IP y número de puerto.
- Ejemplos de número de puertos:
  - HTTP server: 80
  - mail server: 25
- Para enviar un mensaje HTTP al servidor web gaia.cs.umass.edu :
  - IP address: 128.119.245.12
  - port number: 80

Capa de Enlace de Datos

- Objetivo de la **capa de enlace de datos (capa 2)**
  - transformar un medio de transmisión puro en una línea de comunicación que aparezca libre de errores de transmisión.
- **¿Qué problemas de diseño hay que considerar?**
  - Fragmentación de paquetes en **tramas**, cuando un paquete es demasiado grande para ser aceptado por la CED.
    - Transmisión de las tramas de manera secuencial.
  - **Tramas de confirmación de recepción** son usadas cuando el servicio es confiable.
  - **Control de flujo**.
    - Para evitar que un emisor rápido sature a un receptor lento.
  - **Control de acceso a un canal compartido**: se busca manejar y minimizar o evitar colisiones.

Capa de Enlace de Datos

– Control de errores

- **Situación indeseable**: mensajes llegan con errores
- **¿Y esto por qué sucede?**
- **Causa**: medio físico de comunicaciones es imperfecto y ocasiona errores
- **¿Cómo se puede encarar el problema?**
  - ¿Cómo identificar errores?
  - ¿Cómo corregir errores?
- ***¿qué hacer cuando se identifica un mensaje con error?***

Capa Física

- **¿Cuál es el propósito de la capa física (CF)?**
- Transportar un stream de datos de una máquina otra usando medios físicos.
- **¿Cuáles eran los medios de transmisión físicos?**
- **¿Cuál era su función?**
- **¿La CF consiste solo de medios físicos?**
  - No. Los medios físicos **se conectan entre sí** usando dispositivos como codecs, modems, multiplexores, demultiplexores, etc.

Capa Física: medios físicos

- bit: se propaga entre pares de transmisor/receptor
- Enlace físico: lo que yace entre el transmisor & receptor
- Medios guiados:
  - Las señales se propagan en medios sólidos: copper, fiber, coax
- Medios no guiados:
  - Las señales se propagan libremente, e.g., radio

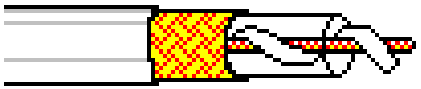
Par trenzado (TP)

- 2 cables de cobre aislados
  - Category 5: 100 Mbps, 1 Gbps Ethernet
  - Category 6: 10Gbps

Capa Física: medios físicos

coaxial cable:

- 2 conductores concéntricos de cobre
- bidireccionales
- broadband:
  - Múltiples canales en el cable



Cable de fibra óptica:

- Fibra de vidrio que transporta pulsos de luz, cada pulso es un bit
- Operan a alta velocidad:
  - high-speed point-to-point transmission (e.g., 10's-100's Gbps transmission rate)
- Baja tasa de errores:
  - repeaters spaced far apart
  - immune to electromagnetic noise

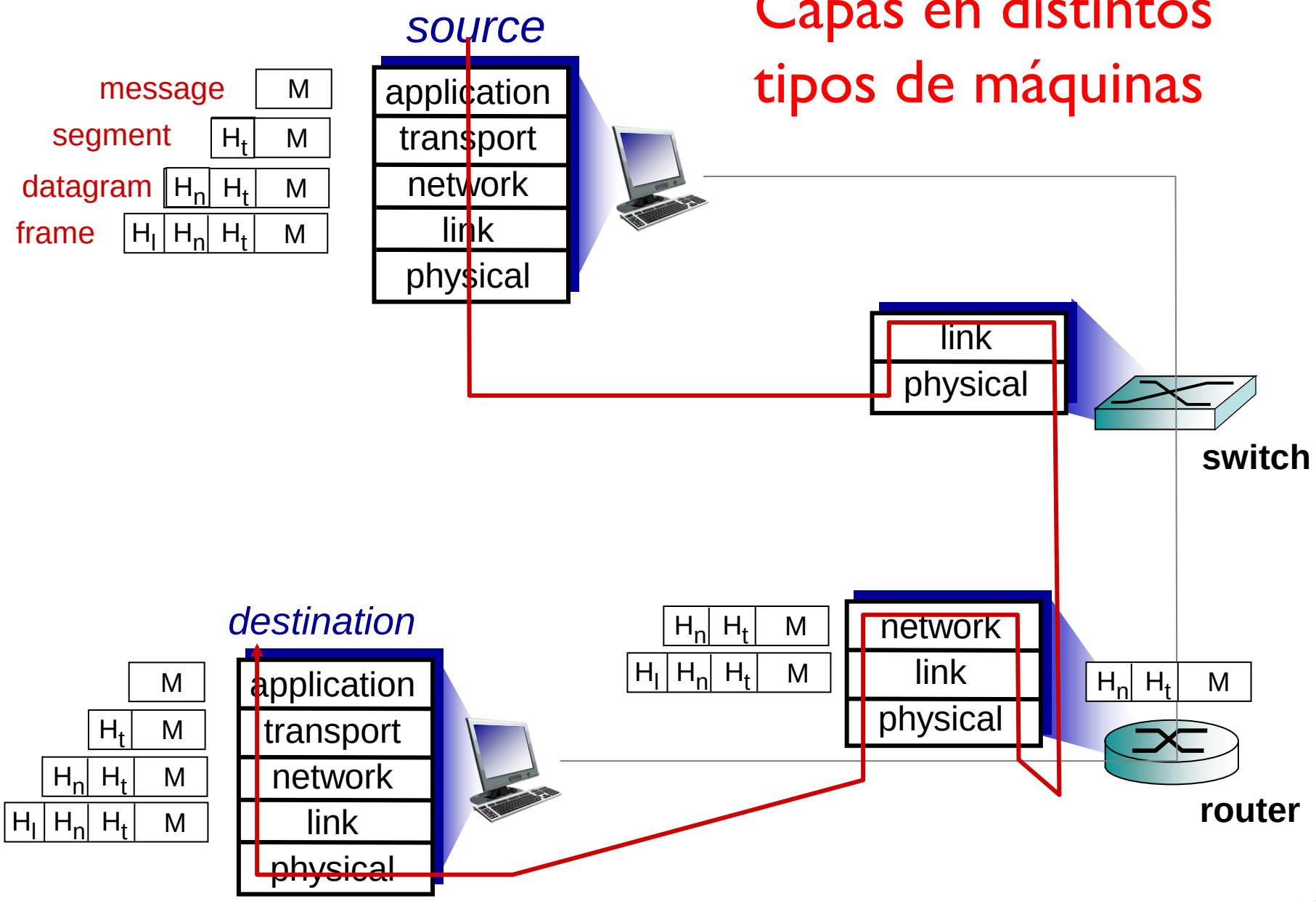
Capa Física: medios físicos

- Radio: signal carried in electromagnetic spectrum
- no physical "wire"
- bidirectional
- propagation environment effects:
  - reflection
  - obstruction by objects
  - interference

radio link types:

- terrestrial microwave
  - e.g. up to 45 Mbps channels
- LAN (e.g., WiFi)
  - 54 Mbps
- wide-area (e.g., cellular)
  - 4G cellular: ~ 10 Mbps
- satellite
  - Kbps to 45Mbps channel (or multiple smaller channels)
  - 270 msec end-end delay
  - geosynchronous versus low altitude

Capas en distintos tipos de máquinas



Protocolos IoT

- **802.15.4 – LR WPAN**: es una colección de estándares para redes de área personal de tasa de transferencia baja (LR-WPANs).
- **6LoWPAN**: (IPv6 over Low Power Wireless Personal Area Networks) trae el protocolo IP a los dispositivos de baja potencia que tienen capacidad de procesamiento limitada.
- **CoAP**: CoAP es un protocolo para usarse en dispositivos de internet restringidos en recursos (p.ej: nodos de redes de sensores inalámbricas).
- **Websocket**: WebSocket se basa en TCP y permite streams de mensages a ser enviados en ambos sentidos entre cliente y servidor, mientras se mantiene la conexión TCP abierta.
- **DDS** (Data distribution service): es un middleware centrado en datos para la comunicación de dispositivo-a-dispositivo or máquina-a-máquina.
- **XMPP** (Extensible Messaging and Presence Protocol) es un protocolo para comunicación de tiempo real y streaming de datos XML entre entidades de red. XMPP soporta caminos de comunicación client-to-server y server-to-server.

Sistemas Operativos de Redes

- **Metas**:
  1. Comprender la arquitectura de los SOR.
  2. Entender cómo funcionan las capas para el envío de mensajes.
  3. Comprender **problemas de diseño** a resolver en distintas capas.
  4. Estudiar los distintos **tipos de capas**
  5. Comprender el SOR modelo TCP/IP
  6. Comprender el **modelo híbrido** de SOR que estudiaremos en la materia.

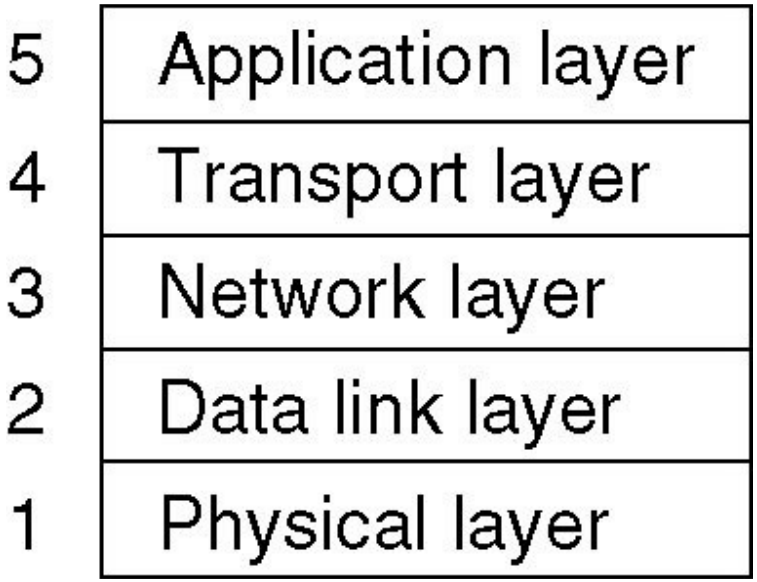
Críticas al modelo de referencia TCP/IP

- **Problemas**:
  - No se distingue entre servicio e interfaz.
    - Se quiere comunicar con aplicaciones por medio de direcciones físicas de interfaces. No hay dirección de nodo o aplicación.
  - No es un modelo general: no esta ajustado para describir ninguna pila de protocolos mas que TCP/IP.
  - No se mencionan las capas físicas y de enlace de datos
  - Protocolos altamente entrincherados y difíciles de remplazar.

Sistemas Operativos de Redes

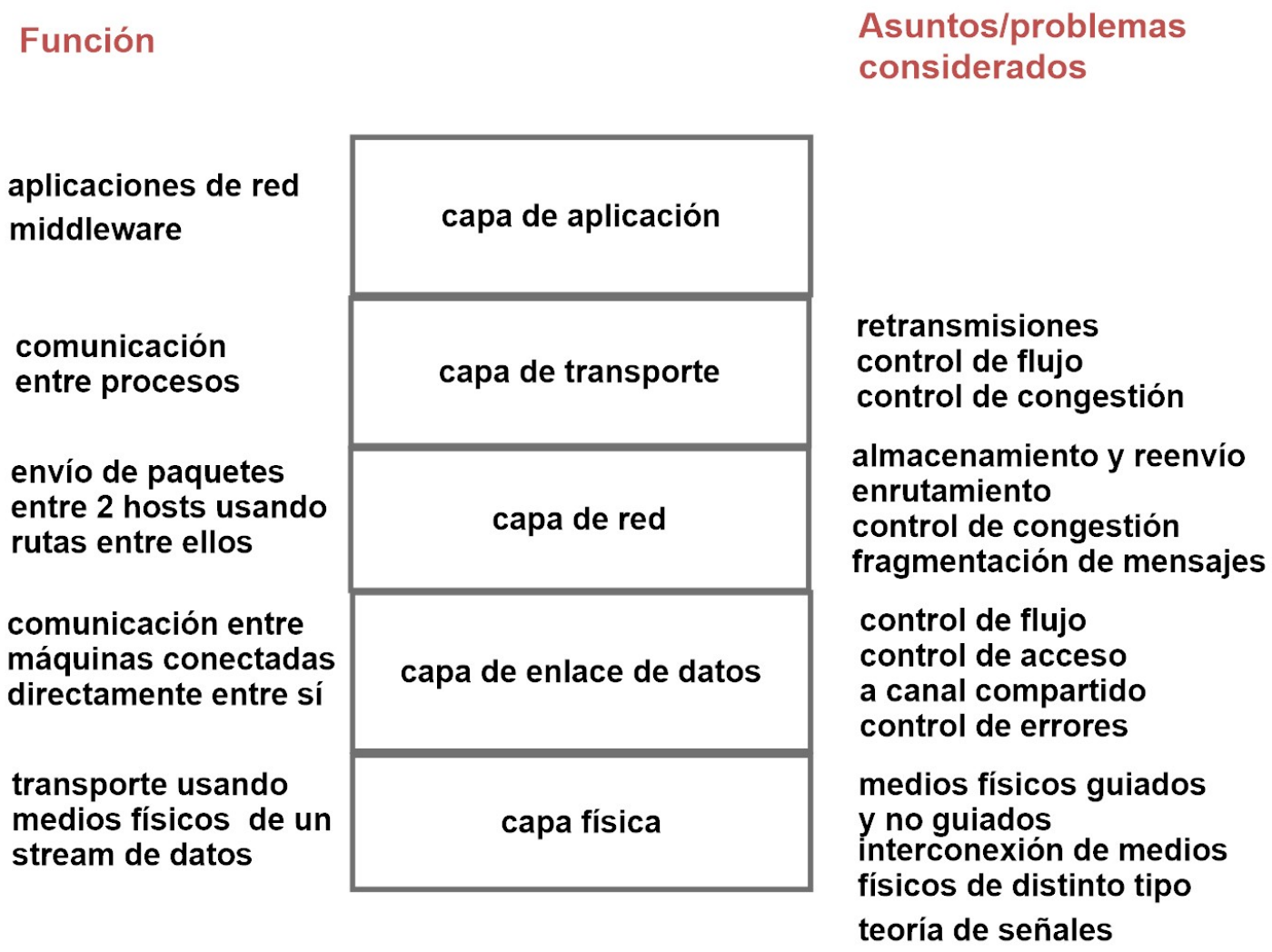
- **Metas**:
  1. Comprender la arquitectura de los SOR.
  2. Entender cómo funcionan las capas para el envío de mensajes.
  3. Comprender **problemas de diseño** a resolver en distintas capas.
  4. Estudiar los distintos **tipos de capas**
  5. Comprender el **modelo híbrido** de SOR que estudiaremos en la materia.

Modelo Híbrido



- En este curso usaremos un modelo híbrido con capas:
  - Física, de enlace de datos, de red, de transporte y de aplicación.
- Nos concentraremos principalmente en el modelo TCP/IP para las capas de red y de transporte.
- **Orden a seguir en la materia: Top_down.**
  - capa de aplicación -> capa de transporte -> capa de red-> capa de enlace de datos -> capa física.

Modelo Híbrido



Metas de la introducción

- **Agenda**:

1. Comprender los distintos **tipos de redes de computadora**.
2. Entender la arquitectura de los **sistemas operativos de redes (SOR).**
3. Aprender fundamentos sobre cómputo en la nube
4. Entender algunas convenciones a respetar en la materia.

Cómputo en la Nube (Cloud)

- **Nube** se refiere a una red pública, privada o híbrida que proporciona **servicios remotos**
  - Permite la manipulación, configuración y acceso a recursos de hardware y software de forma remota.
    - Hay data centers – centros de recursos - a los que se puede acceder.

Cómputo en la Nube (Cloud)

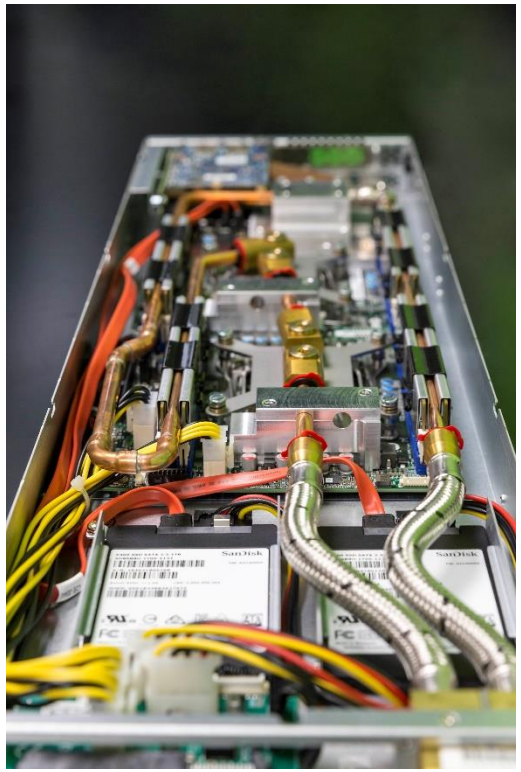
- Hay 3 familias de recursos:
  - **Recursos de procesamiento**: Se puede acceder a:
    - Aplicaciones (P.ej. Google Docs, Gmail)
    - Máquinas virtuales (se las puede acceder por medio de consolas o escritorio remoto)
    - Contenedores: se explica más adelante (p.ej: Dockers)
  - **Recursos de almacenamiento**:
    - Almacenamiento de archivos: (p.ej. dropbox)
    - Almacenamiento de bloques: memoria de sistema asociada a las máquinas virtuales.
  - **Recursos de infraestructura**: combinación de elementos de procesamiento y almacenamiento conectados en una red interna virtual (dentro de un data center).

Cómputo en la Nube (Cloud)

- Datacenters:

"Racks" cajas donde se almacenan servidores

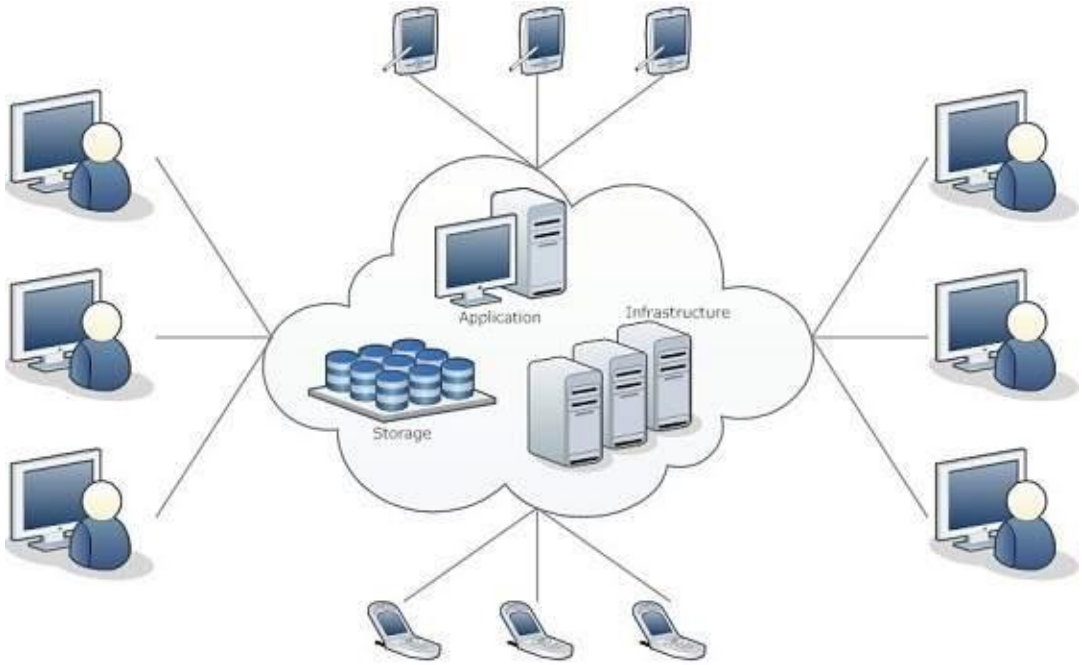
"Patchera" donde se conectan los servidores (dentro de racks)



"Blades" donde está el hardware de los servidores (de ahí salen cables a patcheras)

Cómputo en la Nube (Cloud)

- **Cómo se relaciona esto con redes?**



Cómputo en la Nube (Cloud)

Participantes en la nube:

- Red consumidora de la nube: por ejemplo dentro de compañía.
- Usuarios externos.
- Red proveedora de la nube.
- ISPs

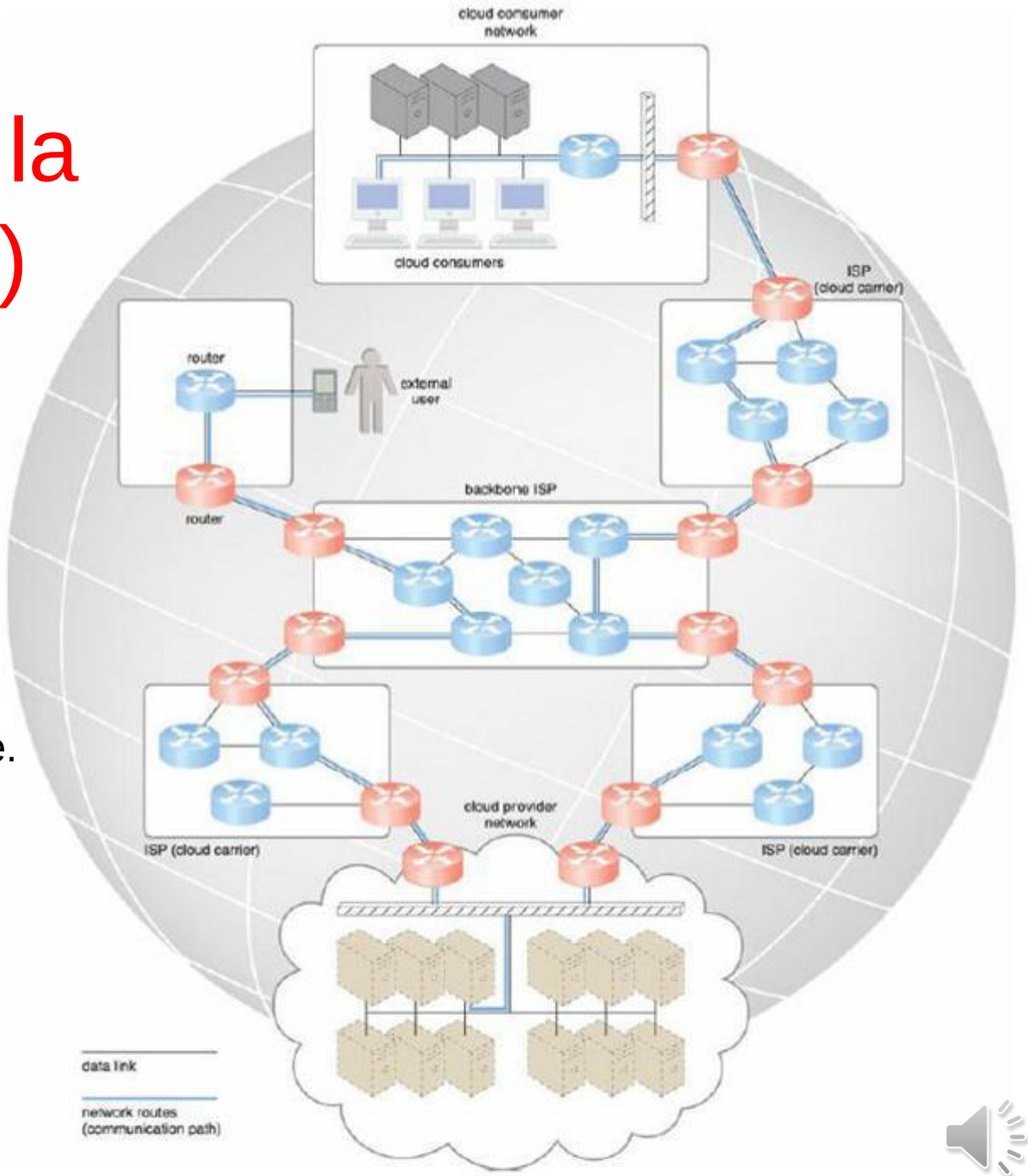


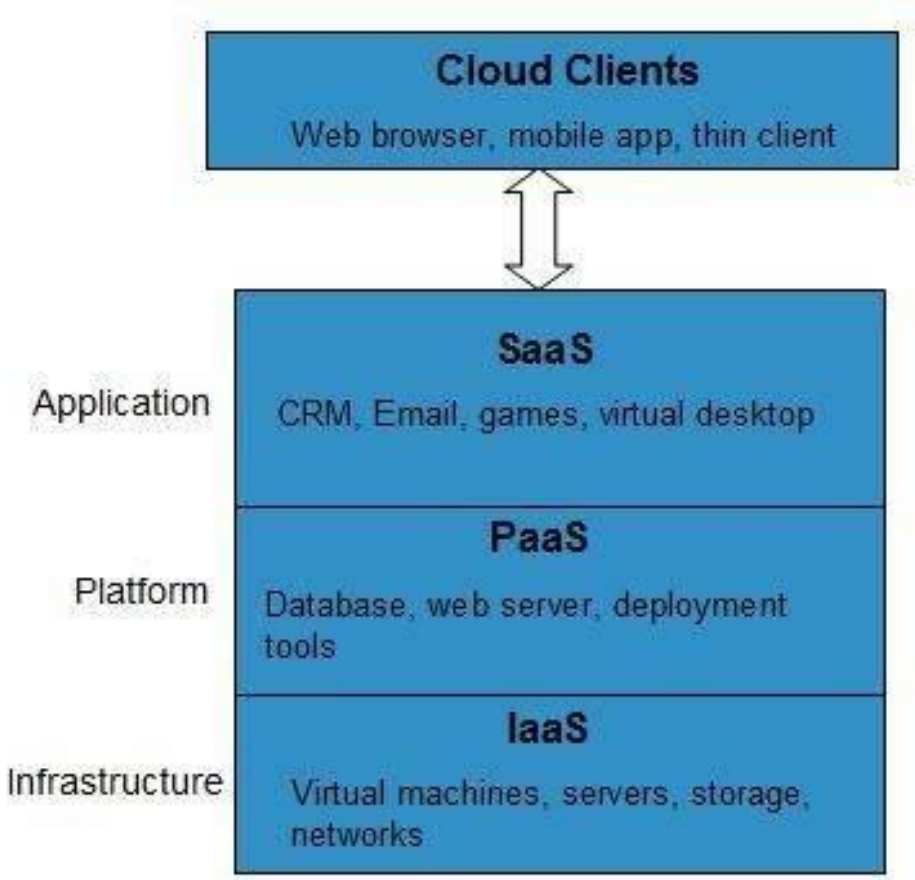
Figure 5.1 Messages travel over dynamic network routes in this ISP internetworking configuration.

Cómputo en la Nube (Cloud)

- **Cómo se relaciona esto con redes?**
- De varias maneras:
  - 1) El **cliente** se conecta a los data centers por medio de redes TCP/IP (Internet), bajo el paradigma cliente-servidor (Se muestra en figura anterior).
  - 2) Dentro de los data-centers, el **hardware** de los servidores se conecta en red (“blades”, “patcheras”, “racks”).
  - 3) Dentro de los data-centers, los recursos de los servidores se abstraen en máquinas **virtuales** o contenedores que se conectan por medio de redes virtuales entre ellos.
    - Hay una red virtual adentro de una red física.

Cómputo en la Nube (Cloud)

- **Servicios**
  - Infrastructure-as–a-Service (**IaaS**)
    - Amazon EC2, W. Azure, Google Compute Engine
  - Platform-as-a-Service (**PaaS**)
    - Heroku, Apache Stratos
  - Software-as-a-Service (**SaaS**)
    - Google Apps, Microsoft Office 365



Cómputo en la Nube (Cloud)

- **Infrastructure-as-a-service (IaaS)**:
  - Ambiente formado por recursos centrados en infraestructura que pueden ser accedidos/manejados vía interfaces basadas en servicios de la nube y en herramientas.
  - Incluye: hardware, red, conectividad, SOs, etc.
  - P.ej: servidor virtual.
  - La administración de los recursos la hace el consumidor.
- **Platform-as-a-service (PaaS)**:
  - Ambiente predefinido listo para usarse que se compone de recursos IT configurados y desplegados.
  - El cliente se libra de la carga administrativa de setear y mantener la infraestructura de recursos IT provistos.
  - P.ej: Google App Engine ofrece un entorno Java y un entorno Python.
  - P.ej: product application server + DBMS platforms.

Cómputo en la Nube (Cloud)

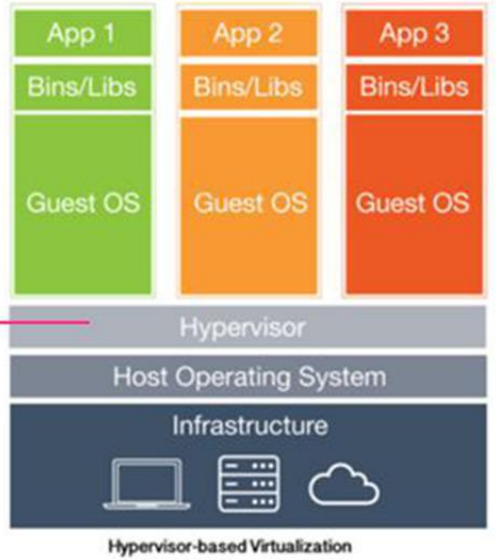
- **Software-as-a-service:**
  - un programa de software posicionado como un recurso compartido en la nube es hecho disponible como un producto.
  - El consumidor de la nube tiene un control administrativo muy limitado sobre el programa de software. Usa y configura el servicio.
  - P.ej: MS Office 365, Google Apps, etc.

Cloud Delivery Model	Common Cloud Consumer Activities	Common Cloud Provider Activities
SaaS	uses and configures cloud service	implements, manages, and maintains cloud service monitors usage by cloud consumers
PaaS	develops, tests, deploys, and manages cloud services and cloud-based solutions	pre-configures platform and provisions underlying infrastructure, middleware, and other needed IT resources, as necessary monitors usage by cloud consumers
IaaS	sets up and configures bare infrastructure, and installs, manages, and monitors any needed software	provisions and manages the physical processing, storage, networking, and hosting required monitors usage by cloud consumers

Cómputo en la Nube (Cloud)

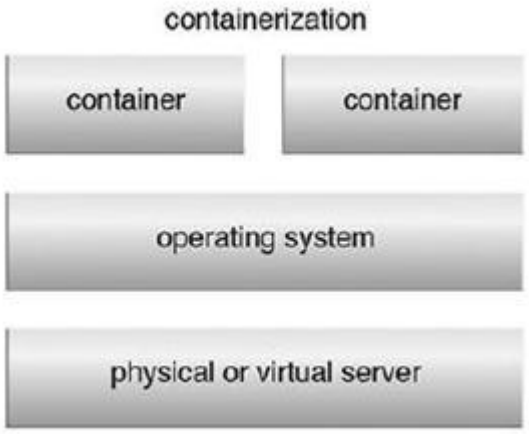
- **Virtualización**:
  - Virtualización es el proceso de convertir un recurso IT físico en un recurso IT virtual.
  - **Servidores virtuales** usan guest OS.
  - La funcionalidad del software de virtualización incluye servicios de sistema relacionados a gestión de máquina virtual; este gestor de máquina virtual se llama **hipervisor**.
  - Tanto guest OS y software de aplicación ejecutando en servidor virtual no son conscientes del proceso de virtualización.

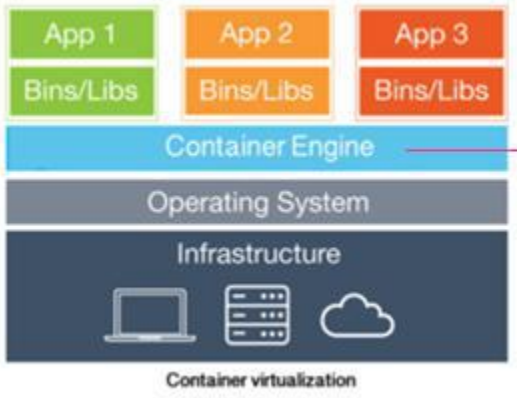
- Virtual Box
- VMware



Cómputo en la Nube (Cloud)

- **Containerización**:
  - Se **empaqueta** el código de la aplicación junto con los archivos de configuración relacionados, librerías y dependencias requeridas para que pueda ejecutar.
  - Este paquete de software o **contenedor** se abstrae del SO y es portable.
  - Las aplicaciones son desplegadas en contenedores. Cada contenedor ejecuta en un proceso.
  - Usar contenedores permite a varios servicios de la nube ejecutarse como un servidor (físico o virtual) único mientras se accede al mismo SO.
  - Un servicio de la nube dentro de un contenedor puede solo ver los contenidos y dispositivos del contenedor asignados al contenedor.
  - Los contenedores son más pequeños en capacidad que una VM y requieren menos tiempo de inicio.





- Docker
- Kubernetes
- Docker compose

Metas de la introducción

- **Agenda**:

1. Comprender los distintos **tipos de redes de computadora**.
2. Entender la arquitectura de los **sistemas operativos de redes (SOR).**
3. Aprender fundamentos sobre **cómputo en la nube**
4. **Entender algunas convenciones a respetar en la materia.**

Convenciones a respetar

- B mayúscula = 1 byte = 8 bits (= 2^3 bits)
- 1KB = 2^10 B = 1024 B (=2^13 bits = 8192 bits) - **kibi**byte
- 1MB = 2^20 B = 1.048.576 B
- 1GB = 2^30 B
- En resumen, para los casos anteriores se usan potencias de 2 junto con bytes.
- En cambio, b minúscula = 1 bit
- 1Kb = 10^3 b = 1000 b – **Kilo** bit
- 1Mb = 10^6 b = 1.000.000 b – **Mega** bit
- 1Gb = 10^9 b = 1000.000.000 b – **Giga** bit
- En resumen, se usan potencias de 10 junto con bits.
- Para expresar velocidades de transmisión:
  - 1Kbps = 1000 bits por segundo
  - 10Mbps = 10^6 bps = 10.000.000 bits por segundo
  - 10Gbps = 10^9 bps

Convenciones a respetar

Exp.	Explicit	Prefix	Exp.	Explicit	Prefix
10^{-3}	0.001	milli	10^{3}	1,000	Kilo
10^{-6}	0.000001	micro	10^{6}	1,000,000	Mega
10^{-9}	0.000000001	nano	10^{9}	1,000,000,000	Giga
10^{-12}	0.000000000001	pico	10^{12}	1,000,000,000,000	Tera

Las unidades de medida pequeñas pueden usarse para medir tiempo
Los unidades de medida grandes pueden usarse para expresar cantidades de bits

Misión

- **Misión de la materia**:
  - Adquirir habilidades con el fin de poder:
    - comprender, evaluar, usar: redes, SOR, middlewares y aplicaciones de red;
    - desarrollar aplicaciones de red.
  - Comprender se refiere a conceptos, principios, problemas y protocolos.
    - Con esta finalidad se usan ejercicios de razonamiento y cuentas que usan matemática elemental.
    - Además se consideran preguntas teóricas.
    - El alumno necesitará poder leer y entender protocolos y aplicaciones de red no enseñados en clase (habrá tareas de este tipo).
    - Puede haber algunos labs que ayuden a comprender en profundidad algunos protocolos.
  - Evaluar se refiere a: comparar, ver semejanzas y diferencias entre distintas alternativas de solución a un problema.
    - Incluye poder decidir cuál solución adoptar para un problema.