Introducción a Capa de Aplicación

Redes y Comunicaciones

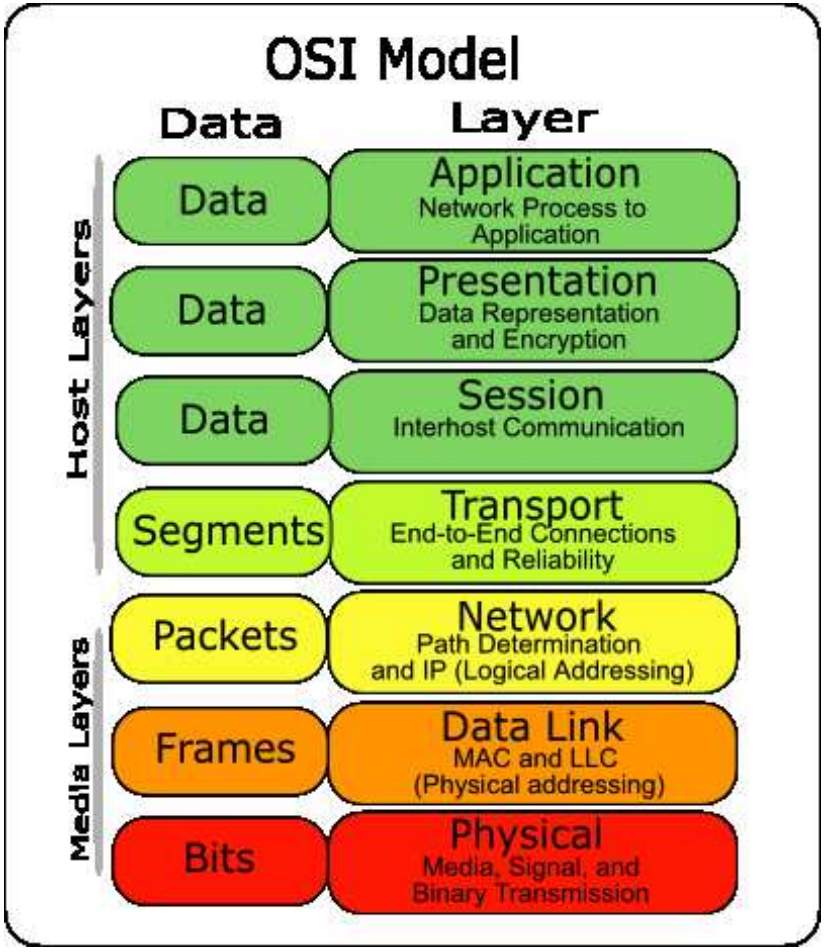
Funciones de la Capa de Aplicación

- Provee servicios de comunicación a los usuarios (Capa 8 ;)) y a las aplicaciones, incluye las aplicaciones mismas.
- Existe modelo de comunicación Machine to machine (M2M), no hay usuarios (personas).
- Interfaz con el usuario -User Interface (UI)- u otras aplicaciones/servicios.
- Las aplicaciones que usan la red pertenecen a esta capa.
- Los protocolos que implementan las aplicaciones también.
- Existen aplicaciones que NO son de red que deben trabajar con aplicaciones/servicios para lograr acceso a la red.

Componentes de la Capa de Aplicación

- Elementos de capa de aplicación: Programas que corren en (diferentes) plataformas y se comunican entre sí y los protocolos que implementan.
- Las aplicaciones en la mayoría de los casos corren en los nodos finales(end-systems), no en el núcleo de la red. (más fácil el desarrollo y uso), siguen principio end-2-end.
- Qué cubre?: Se verá el enfoque orientado a Internet, en el cual la capa de Aplicación integra:
  - Capa de Aplicación propiamente dicha del modelo OSI.
  - Capa de Presentación del modelo OSI.
  - Capa de Sesión del modelo OSI.

Modelo OSI


OSI Model
Data
Layer
Host Layers
Media Layers
Data
Application
Network Process to Application
Data
Presentation
Data Representation and Encryption
Data
Session
Interhost Communication
Segments
Transport
End-to-End Connections and Reliability
Packets
Network
Path Determination and IP (Logical Addressing)
Frames
Data Link
MAC and LLC (Physical addressing)
Bits
Physical
Media, Signal, and Binary Transmission

Capa de Sesión

- Administra las conversaciones/diálogos entre las aplicaciones.
- Podría proveer mecanismos transaccionales o de sincronización: COMMIT, CHECKPOINT, ROLLBACK.
- Maneja actividad.
- Informar Excepciones.
- Ejemplo concreto RPC (Remote Procedure Call) en NFS.
- A menudo las facilidades del lenguaje de acceso a bases de datos: SQL podría verse como un ejemplo.
- Integrada en las aplicaciones de red mismas.
- Podría estar ausente.

Capa de Sesión (Consideraciones)

- Básicamente la Capa de sesión se podría ver como un invento de la ISO.
- Ninguna de las redes existentes tenían esta capa (salvo algunos servicios de modelo OSI aparecen en SNA).
- La capa de sesión es muy “delgada”, con relativamente pocas características.
- En general no se utiliza.

Capa de Presentación

- Conversión y codificación de datos a codificaciones comunes, ej: ASCII, EBDIC, charset ISO-8859-1, UTF-8, Unicode16.
- Compresión y descompresión de datos.
- Cifrado y de-cifrado de datos.
- Define formatos y algoritmos para esto: JPEG, MPEG, LZW, AES, DES, IDEA.
- Ejemplo concreto: XDR (External Data Representation) en NFS.
- Integrada en las aplicaciones de red mismas.

Capa de Aplicación

- Define el formato de los mensajes. Existen protocolos que trabajan de forma binaria, por ejemplo usando ASN y otros en forma textual ASCII como HTTP.
- Define la semántica de cada uno de los mensajes.
- Define como debe ser el diálogo (intercambio de mensajes). Que mensajes se deben intercambiar.
- Ejemplo concreto: Protocolo HTTP y sus implementaciones mediante servidores WEB y browsers (navegadores).

Capa de Aplicación

- Define el formato de los mensajes. Existen protocolos que trabajan de forma binaria, por ejemplo usando ASN y otros en forma textual ASCII como HTTP.
- Define la semántica de cada uno de los mensajes.
- Define como debe ser el diálogo (intercambio de mensajes). Que mensajes se deben intercambiar.
- Ejemplo concreto: Protocolo HTTP y sus implementaciones mediante servidores WEB y browsers (navegadores).

Modelos de Comunicación de Aplicaciones

- Modelo Mainframe (dumb client).
- Modelo Cliente/Servidor.
- Modelo Peer to Peer (P2P).
- Modelo Híbrido.

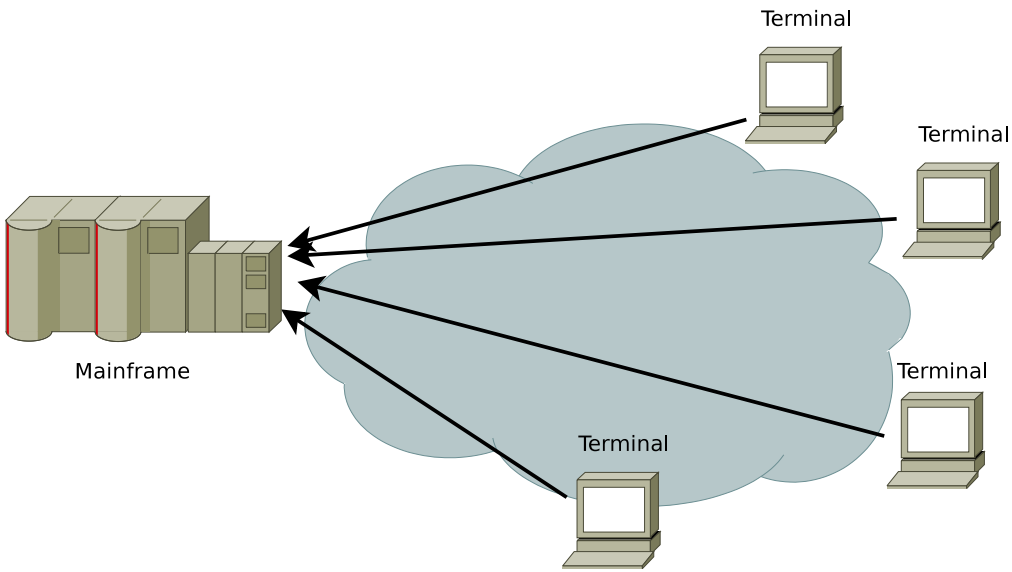
Modelo de Mainframe Centralizado

- Modelo de Carga concentrada.
- El cliente es “tonto” (dumb) solo corre la comunicación y la interfaz física con el usuario (ej. terminal).
- El servidor pone todo el procesamiento.
- Modelo antiguo que resurge con thin-clients.
- Modelo puro: el mainframe es el que decide cuando le da el control al cliente
- Modelo puro: el mainframe maneja el diálogo de las comunicaciones.
- Cliente ejecuta en el mainframe.
- Ejemplo: Sistema SNA con Mainframe IBM S/370 y terminales

3270 (las terminales verdes del antiguo sistema de alumnos !!!).

- Servidor de Terminales: X11, VNC, Cytrix Metaframe, VMWARE PCoIP, LTS, Virtualización.

Modelo de Mainframe Centralizado (Cont'd)

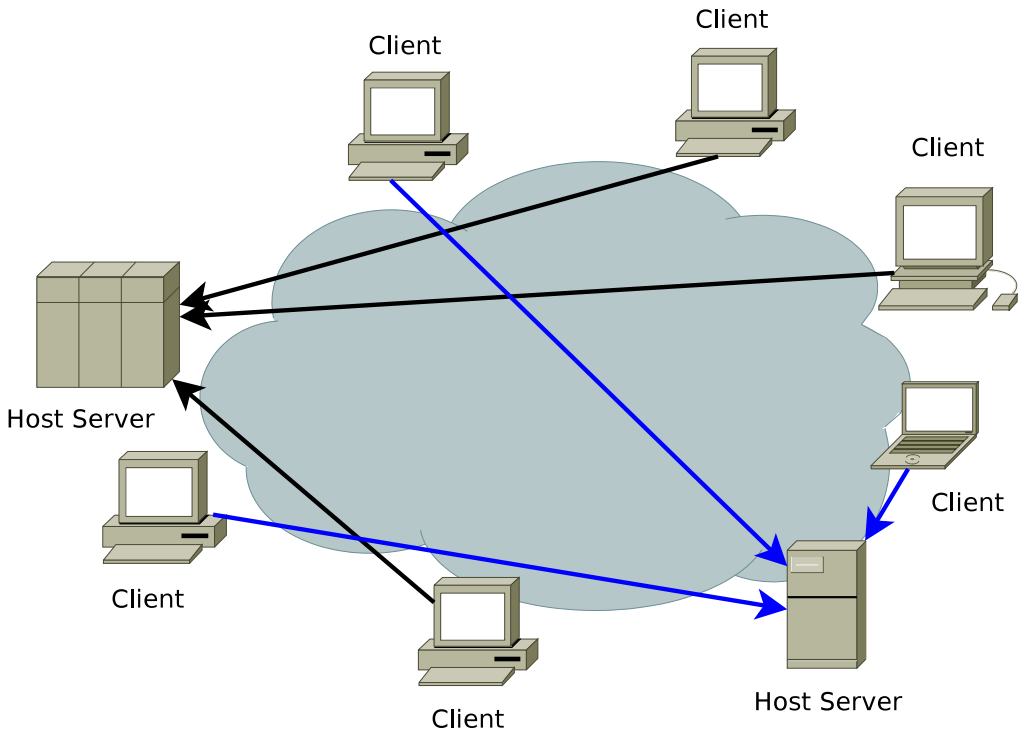


Modelo de Mainframe Centralizado (Cont'd)

Modelo de Cliente/Servidor

- Modelo de Carga compartida.
- Idea inicial: el cliente pone procesamiento de interfaz.
- El servidor pone el resto del procesamiento.
- Existen modelos en varios tiers (2 tiers, 3 tier o multi-tier).
- El servidor corre servicio esperando de forma pasiva la conexión.
- Cliente se conectan al servidor y se comunican a través de este.
- Ejemplo: File Server vía NFS o FTP.
- Moldeo asimétrico 1 a N, M a N (donde M$<$N).

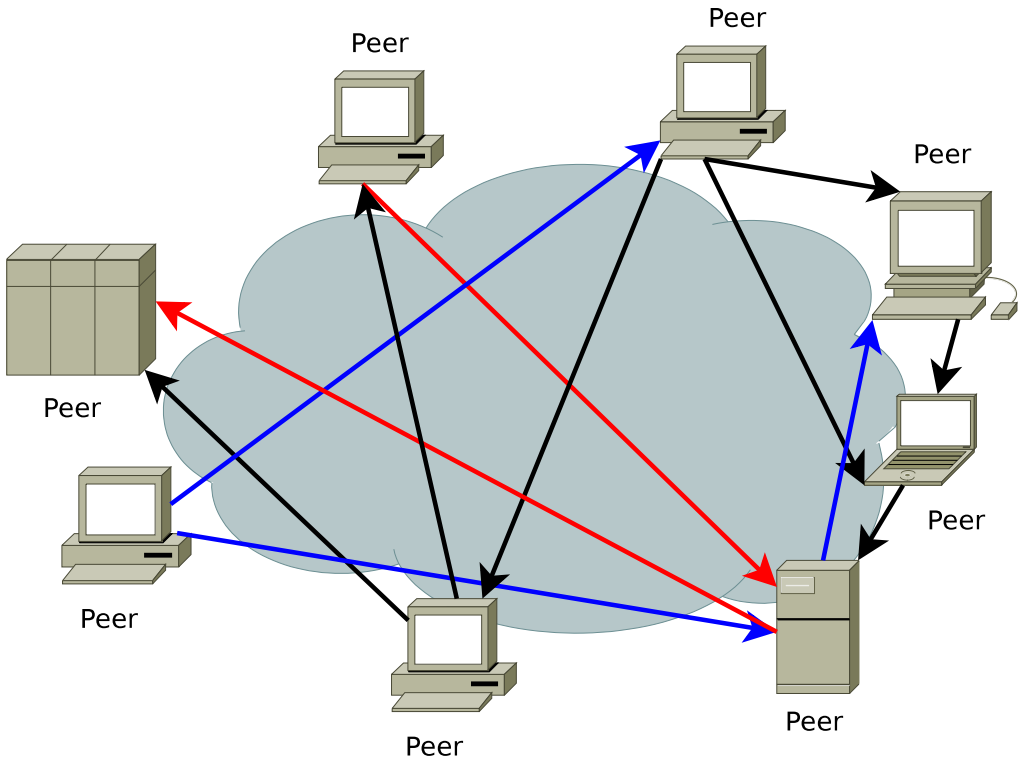
Modelo de Cliente/Servidor (Cont'd)



Modelo de Peer-to-Peer

- Modelo de Carga completamente compartida y distribuida.
- Los peers (participantes) pueden cumplir rol de cliente, servidor o ambos en un instante.
- Sistema escalable en cuanto a rendimiento.
- Sistema no escalable en cuanto a administración.
- Ejemplo: redes legadas para compartir archivos: Novell Lite, Microsoft Windows for Workgroup basado en LAN Manager (sobre NetBEUI), Algo más actual: Gnutella, Bittorrent. (servicio de file sharing totalmente P2P)
- Modelo asimétrico N a N.

Modelo de Peer-to-Peer (Cont'd)



Modelo de Peer-to-Peer Híbrido

- Modelo de Carga compartida y distribuida.
- Los peers (participantes) pueden cumplir rol de cliente, servidor o ambos en un instante.
- Existen diferentes tipos de nodos con diferente roles.
- Hay nodos centrales donde se registra la información y al resto de los nodos.
- Sistema escalable en cuanto a rendimiento.
- Sistema más escalable que P2P puro?.
- Ejemplos: eDonkey (y sus variantes aMule, eMule), Napster, IM, Skype.
- Moldeo asimétrico M a N.

BitTorrent

- Protocolo para el intercambio de archivos grandes de forma masiva como peer-to-peer (P2P).
- Desarrollado originalmente por un programador, Bram Cohen (software libre).
- Cuando alguien quiere compartir un archivo genera un archivo `.TORRENT`.
- A diferencia de otros P2P no tiene índices de búsquedas, si existen WEBs que permite localizar archivos: Mininova, Thepiratebay, Ktorrents, TorrentReactor.

BitTorrent (Cont'd)

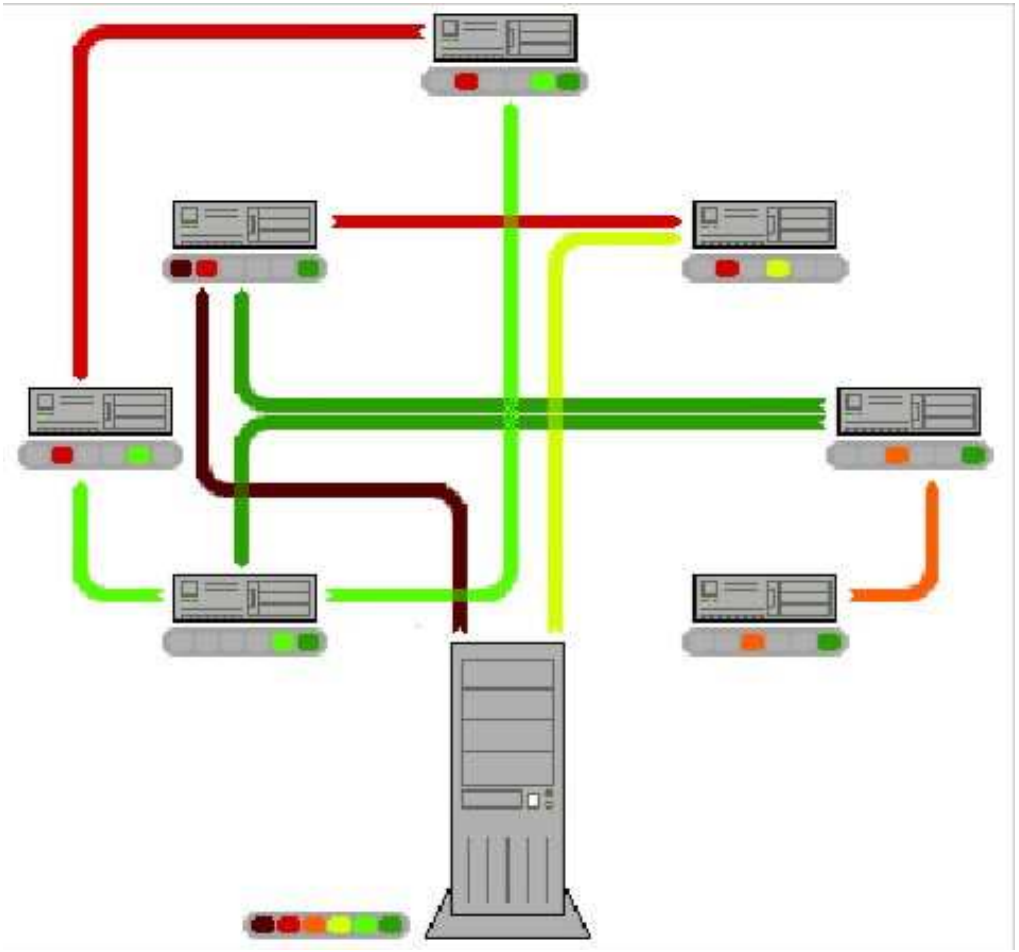
- `.TORRENT`. Dirección del TRACKER para unirse al grupo de PEERs (más información de archivo).
- Desde el TRACKER se obtiene PEERs: SEEDs (contienen archivo completo) y LEECHERs (contienen archivo parcial). El TRACKER se actualiza con el nuevo PEER.
- La comunicación con el TRACKER habitualmente se hace sobre HTTP, o podría ser sobre FTP.
- Los archivos se dividen en chunks, cada uno puede ser descargado de un PEER diferente.
- Se conecta con otros PEERs y comienza la descarga. Cada chunk se comparte con otros PEERs. Utiliza el port 6881 y escanea hacia arriba.

BitTorrent (Cont'd)

- Cuando baja trabaja de forma aleatoria o "rarest first algorithm", trata de bajar los chunks con menos números de copias.
- Por cada archivo puede haber como máximo 4 PEERs remotos descargando. Los peers se rankean, se deja bajar de aquellos que mantienen una buena relación de bajada y subida con el PEER.
- Se seleccionan PEERs aleatoriamente cada determinado tiempo.
- BitTorrent puede trabajar con NAT, si el cliente se conecta (outbound connection) a otro PEER que tiene IP pública y esta conexión se usa para download y upload. Es conveniente habilitar port forwarding.

- Se puede restringir download y upload rate.

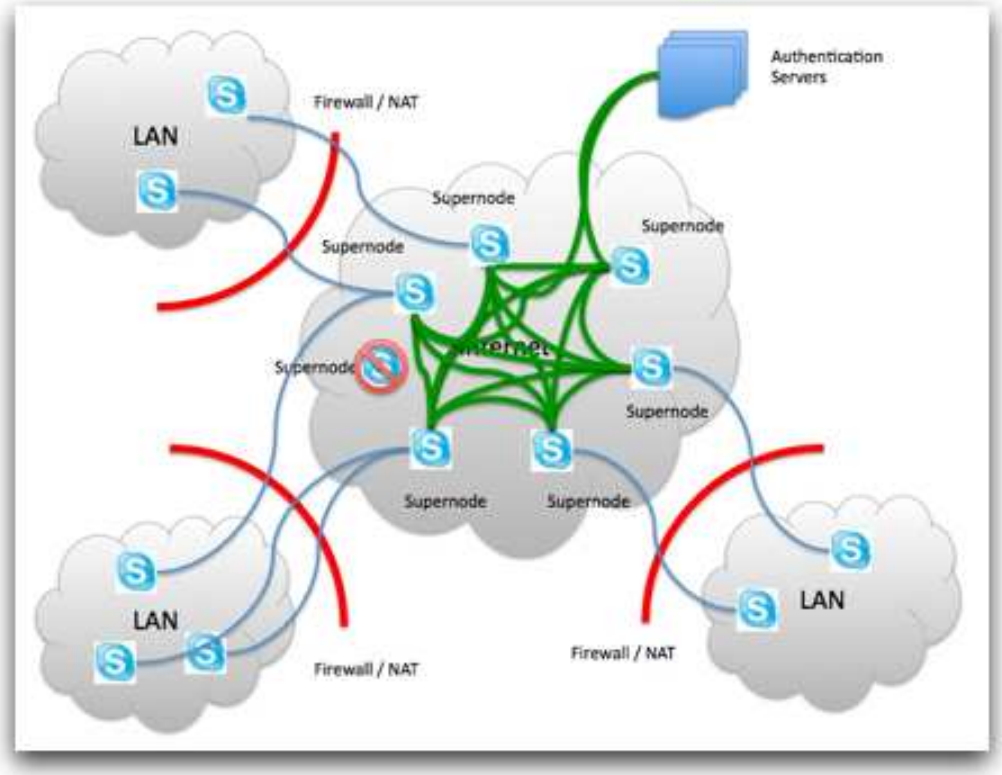
BitTorrent (Cont'd)



Skype

- Protocolo propietario que implementa VoIP.
- Nodos clientes pueden ser solo nodos o Super-nodos.
- Super-nodos actúan como directorios.
- Estos nodos hace de STUN, permitiendo conexión de computadoras detrás de NAT.
- Los super-nodos crean P2P Overlay networks (Una red sobre otra red).
- Corren el software tradicional de los clientes, pero sobre la Internet pública.
- Super-nodos se conectan entre si. Los Super-nodos utilizan servicios externos de autenticación.
- Existen Mega Super-nodos que pertenecen a Skype.

Skype (Cont'd)


Authentication Servers
Firewall / NAT
LAN
Supernode
Supernode
Supernode
Supernode
Supernode
Supernode
Supernode
Supernode
Supernode
LAN
LAN
Firewall / NAT
Firewall / NAT

Requerimientos de Aplicaciones

Aplicación	Pérdidas	Bandwidth	Sensible a Time
file transfer	no	Flexible	no
e-mail	no	Flexible	no
Web documents	no	Flexible	no
real-time audio/video	tolerante	audio: 5kbps-1Mbps video:10kbps-5Mbps	si, 100's msec
stored audio/video	tolerante	Igual al de arriba	si, pocos secs
interactive games	tolerante	Pocos Kbps	si, 100's msec
instant messaging	no	flexible	Si y no

- Cada aplicación puede tener diferentes requerimientos: seguridad, tiempo de respuesta, confiabilidad, optimizar ancho de banda, etc...
- Diferentes Alternativas de Transporte.

- Transport Control Protocol (TCP).
- User Datagram Protocol (UDP).
- Otras alternativas: Stream Control Transmission Protocol (SCTP), Reliable User Datagram Protocol (RUDP).

Direccionamiento de Procesos

- Las aplicaciones son implementadas por los SO como procesos.
- Conceptos de IPC (Inter-Process Communication).
- Para que un proceso reciba un mensaje, éste debe tener un identificador único.
- Identificador de host no suficiente, se agrega identificador de proceso (independiente del SO) número de puerto.
- Servicio de multiplexación provisto por nivel de transporte.
- Accedido mediante la API de **Sockets BSD**, Winsocks de Microsoft o TLI/XTI de AT&T.

Fuentes de Información

- Kurose/Ross: Computer Networking (6th Edition).
- Andrew S. Tanenbaum. Computer Networks (4th Edition).
- Wikipedia http://www.wikipedia.org.
- Bittorrent: http://www.bittorrent.org.
- Understanding Today's Skype Outage: Explaining Supernodes, Dan York, Dec 2010.
- Slides de Kurose/Ross Computer Networking 6ta edición.
- Internet ...