

# CAPA DE APLICACIÓN

Este material está basado en :

- Material preparado como apoyo al texto *Computer Networking: A Top Down Approach Featuring the Internet*, 3<sup>rd</sup> edition. Jim Kurose, Keith Ross Addison-Wesley, July 2004.

# Capa de Aplicación

## Objetivos:

- Aspectos conceptuales y de implementación de los protocolos de aplicación
  - Modelo de servicio de la capa transporte
  - Paradigma cliente-servidor
  - Paradigma peer-to-peer (par-a-par)
- Aprendizaje de protocolos examinando protocolos de aplicación populares
  - HTTP
  - FTP
  - SMTP / POP3 / IMAP
  - DNS

# Algunas aplicaciones de red

- E-mail
- Web
- Mensajería instantánea
- Login remoto
- Compartición de archivos P2P
- Juegos de red multi-usuarios
- Reproducción de clips de video almacenados
- Telefonía Internet
- Conferencias de video en tiempo real
- Computación paralela masiva.

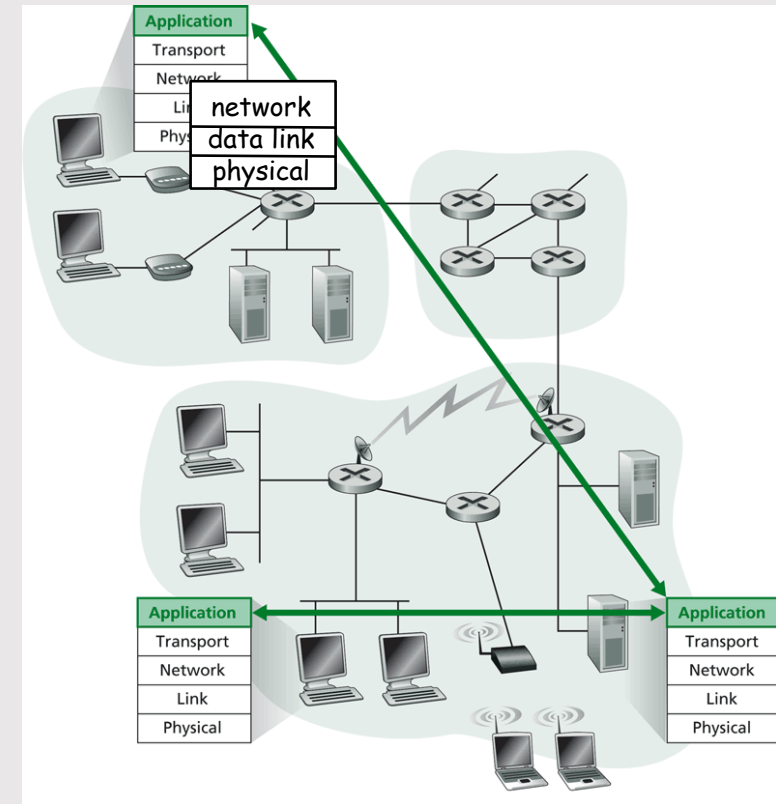
# Características de una aplicación de red

## Son programas que

- Corren en diferentes sistemas terminales.
- Se comunican por la red.
- Ejemplo: Programa de servidor web que se comunica con el programa del navegador web

## No se necesita escribir software para los dispositivos intermedios de red

- Los dispositivos intermedios no ejecutan aplicaciones de usuario
- Las aplicaciones en sistemas terminales permiten desarrollo y despliegue rápido

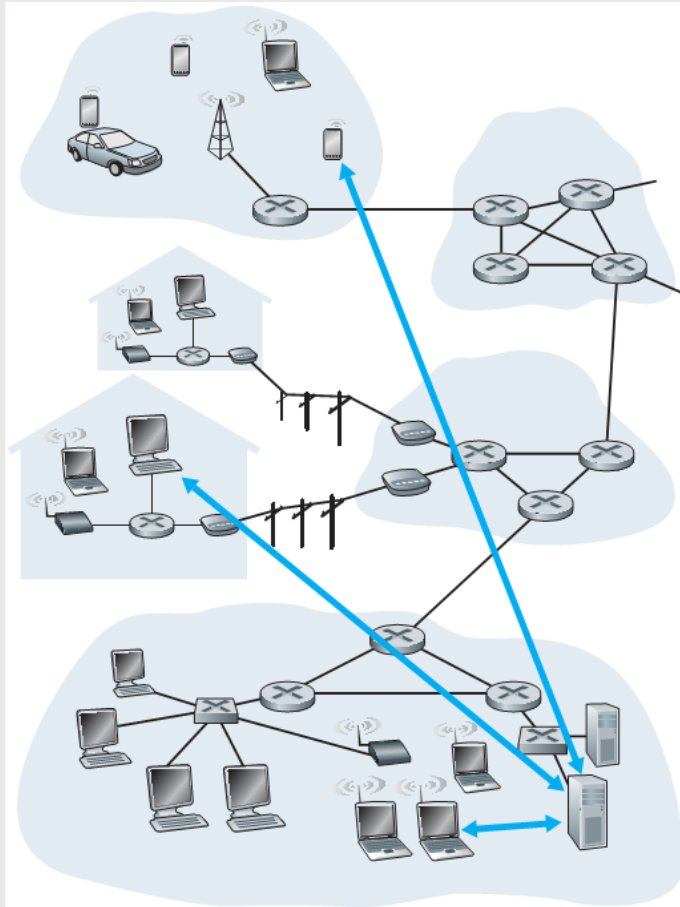


# PRINCIPIOS DE APLICACIONES DE RED

# Arquitecturas de Aplicación

- Las aplicaciones de red pueden utilizar una de las siguientes arquitecturas:
  - Cliente-servidor
  - Peer-to-peer (P2P)
  - Híbridos de cliente-servidor y P2P

# ARQUITECTURA CLIENTE-SERVIDOR



## Servidor:

- Computador siempre encendido
- Dirección IP permanente
- Granja de servidores por escalamiento

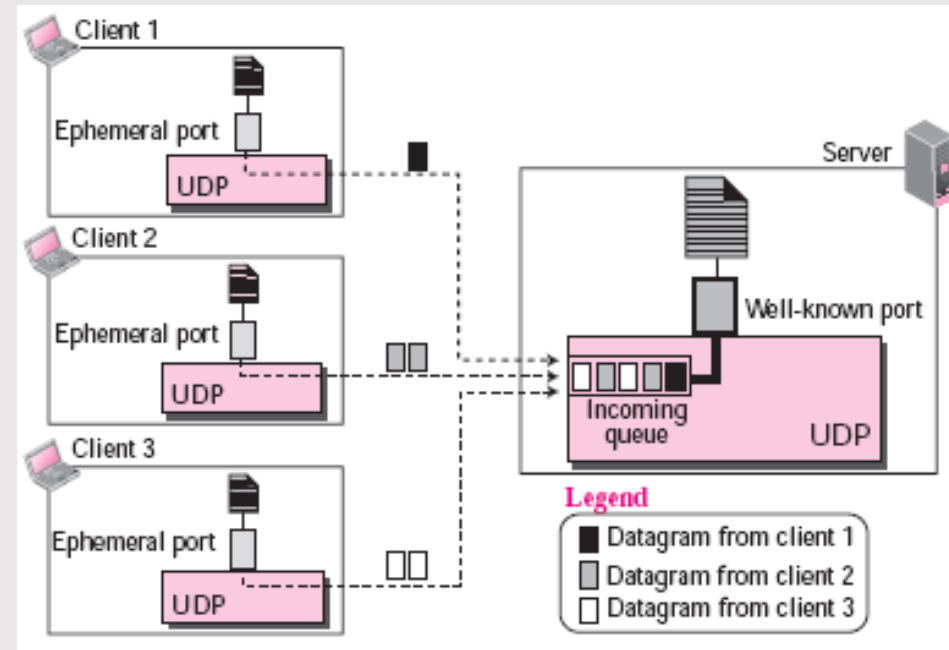
## Cliente:

- Se comunica con servidor
- Puede ser conectado intermitentemente
- Puede tener direcciones IP dinámicas
- No se comunican directamente entre sí (dos clientes puros)

# ARQUITECTURA CLIENTE-SERVIDOR

## Servidor iterativo no orientado a la conexión:

- Procesa una petición a la vez.
- Los datagramas se almacenan en una cola a la espera de ser atendidos en orden de llegada
- El servidor utiliza un único puerto
- Utilizan UDP como protocolo.

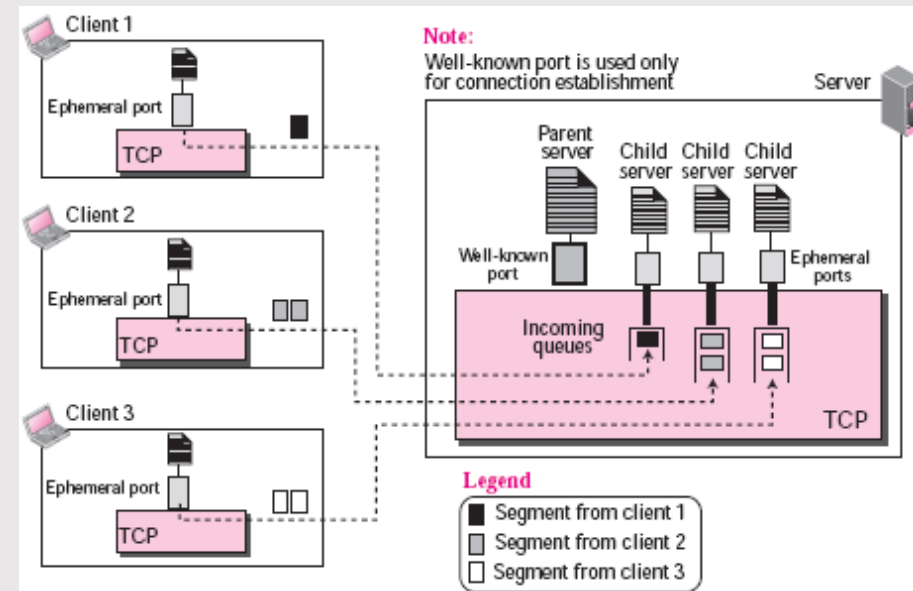




# ARQUITECTURA CLIENTE-SERVIDOR

## Servidor concurrente orientado a la conexión

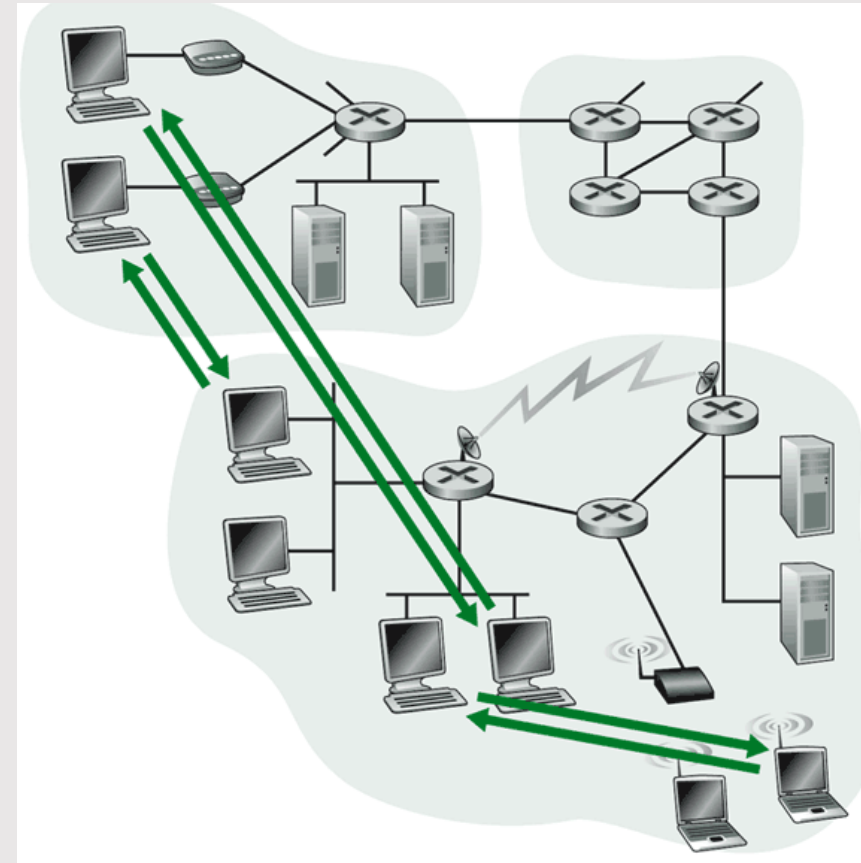
- Puede atender a múltiples clientes a la vez
- Se establece una conexión entre el servidor y cada cliente y esta permanece activa hasta que el flujo completo es procesado y se cierra la conexión.
- Utiliza un puerto bien conocido y crea puertos efímeros para cada conexión.
- Se crea una cola por conexión.



# ARQUITECTURA P2P PURA

- Servidor no siempre encendido
- Sistemas terminales arbitrarios se comunican directamente
- Pares se conectan intermitentemente y cambian sus direcciones IP
- Ejemplo: BitTorrent

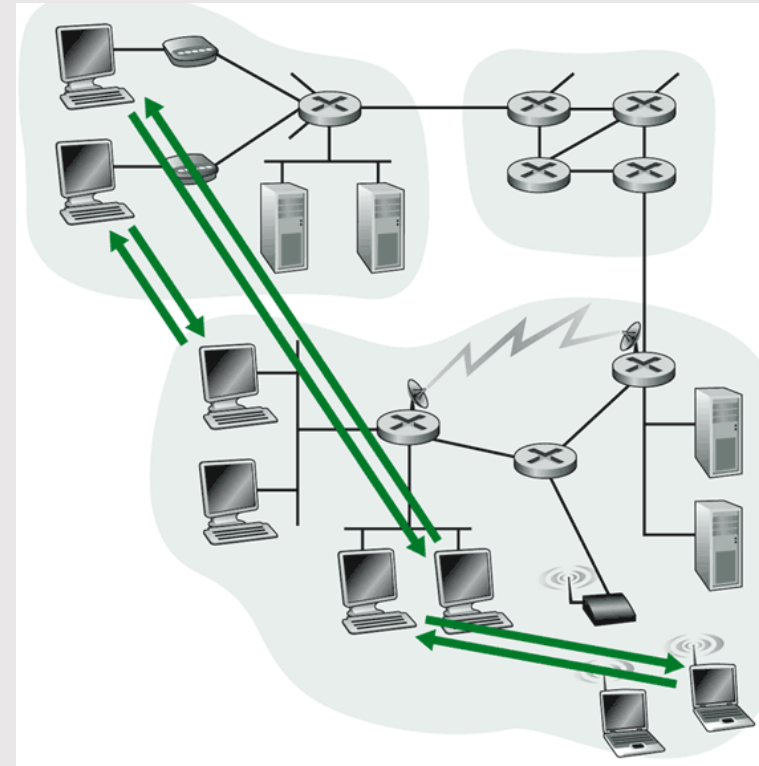
Altamente escalable, pero difícil de administrar



# ARQUITECTURA P2P PURA

## Retos de la arquitectura P2P

- La característica asimétrica de las tecnologías de red domestica (ADSL), hacen poco atractivo para los ISP el uso de esta arquitectura
- Debido a su naturaleza altamente distribuida, la seguridad es un aspecto importante
- Se debe motivar a los usuarios a compartir ancho de banda y otros recursos



# HÍBRIDOS DE CLIENTE-SERVIDOR Y P2P

## Skype

- Aplicación P2P de Voz sobre IP
- Servidor centralizado: búsqueda de direcciones de partes remotas
- Conexión cliente – cliente: directa (no a través del servidor)

## Mensajería Instantánea

- Diálogo entre dos usuarios es P2P
- Servicio centralizado: Detección/localización de presencia de cliente:
  - Usuario registra su dirección IP en un servidor central cuando ingresa al sistema
  - Usuarios contactan servidor central para encontrar las direcciones IP de sus amigos.

# COMUNICACIÓN ENTRE PROCESOS

**Proceso:** programa que corre en un host.

- Dentro de la máquina dos procesos se comunican usando **comunicación entre proceso** (definida por OS).
- Procesos en diferentes hosts se comunican vía intercambio de **mensajes**

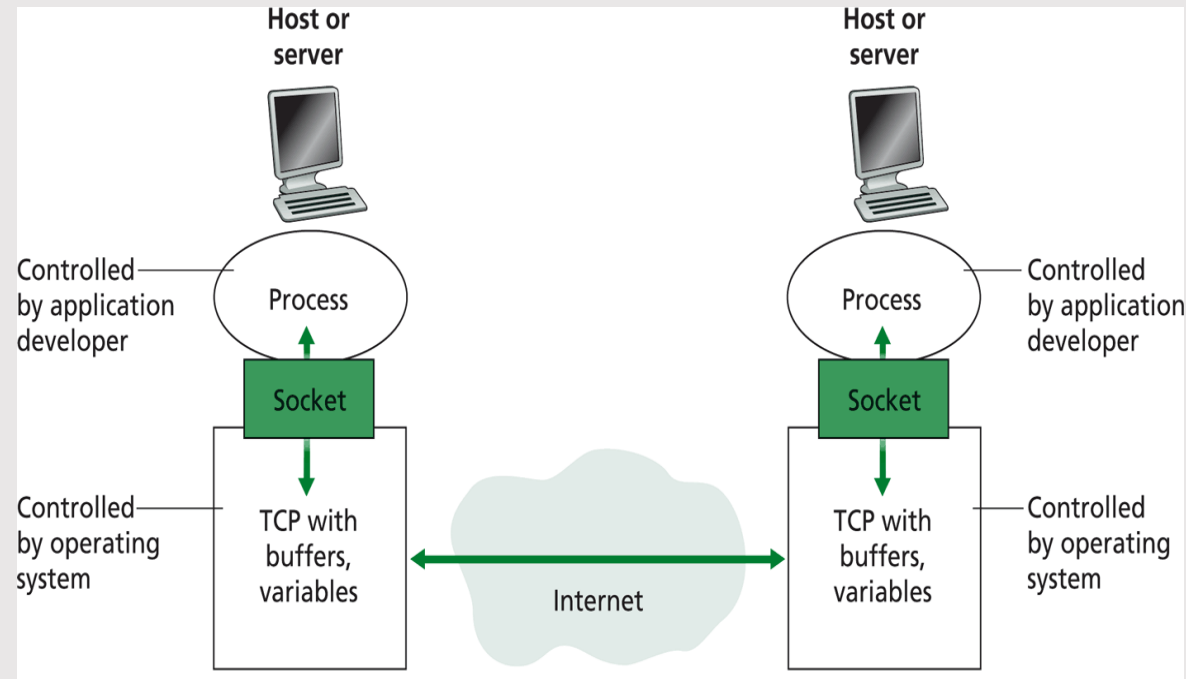
**Proceso Cliente:** proceso que inicia la comunicación

**Proceso Servidor:** proceso que espera por ser contactado

**Nota:** las aplicaciones con arquitectura P2P tienen procesos clientes y procesos servidores

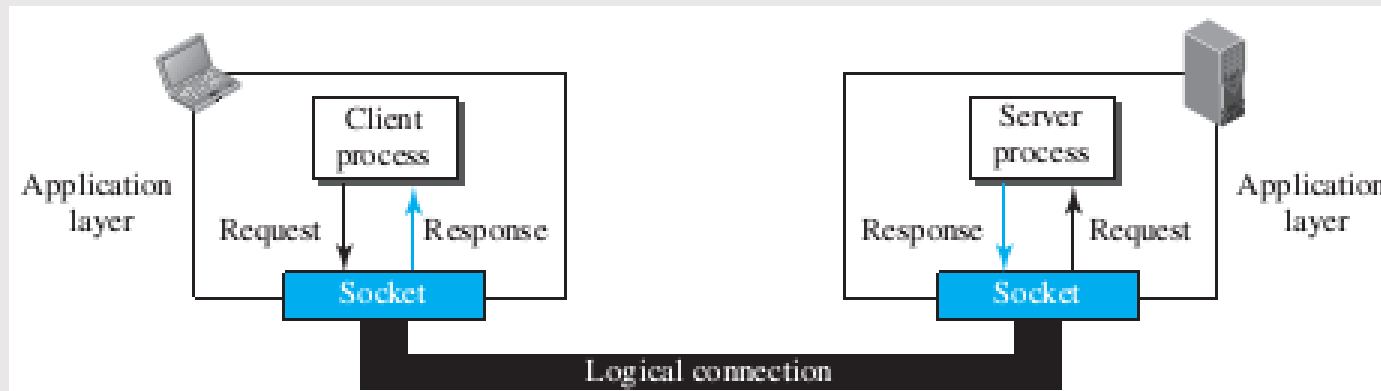
# SOCKETS

Es la interfaz, a través de la cual, los procesos envían y reciben mensajes



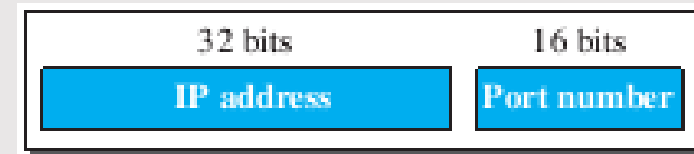
# SOCKETS

- Los socket son análogos a puertas
  - Proceso transmisor saca mensajes por la puerta
  - Proceso transmisor confía en la infraestructura de transporte al otro lado de la puerta la cual lleva los mensajes al socket en el proceso receptor



# DIRECCIONAMIENTO DE PROCESOS

- Para que un proceso reciba un mensaje, éste debe tener un identificador
- Un host tiene una dirección IP única de 32 bits.
- El identificador incluye la **dirección IP** y un **número de puerto** asociado con el proceso en el host.
- Ejemplo de números de puertos:
  - Servidor HTTP: 80
  - Servidor de Mail: 25
- Para enviar un mensaje HTTP al servidor [www.concytec.gob.pe](http://www.concytec.gob.pe)
  - **Dirección IP**: 190.12.64.4
  - **Número de puerto**: 80





# LOS PROTOCOLOS DE CAPA DE APLICACIÓN DEFINEN

- Tipos de mensajes intercambiados
  - e.g., mensajes de requerimiento y respuesta
- Sintaxis de los tipos de mensajes:
  - Qué campos en los mensajes y cómo éstos son delineados.
- Semántica de los campos
  - Significado de la información en los campos
- Reglas sobre cuándo y cómo los procesos envían y responden a mensajes

## Protocolos de dominio público:

- Definidos en RFCs
- Permite interoperabilidad
- Eg: HTTP, SMTP

## Protocolos propietarios:

- Eg: KaZaA, Skype

# SERVICIOS DE TRANSPORTE REQUERIDOS POR UNA APLICACIÓN

## Transferencia de Datos Fiable

- Algunas aplicaciones (e.g., audio) pueden tolerar pérdida
- otras (e.g., transferencia de archivos, telnet) requieren transferencia 100% confiable

## Retardo

- Algunas Aplicaciones (e.g., Telefonía Internet, juegos interactivos) requieren bajo retardo para ser “efectivas”

## Ancho de banda

- Algunas aplicaciones (e.g., multimedia) requieren una cantidad mínima de ancho de banda para ser “efectivas”
- Otras (“aplicaciones elásticas”) hacen uso del ancho de banda que obtengan

## Seguridad

- Algunas Aplicaciones (e.g., Mensajería) requieren diversos tipos de servicios de seguridad como confidencialidad, integridad y autenticación

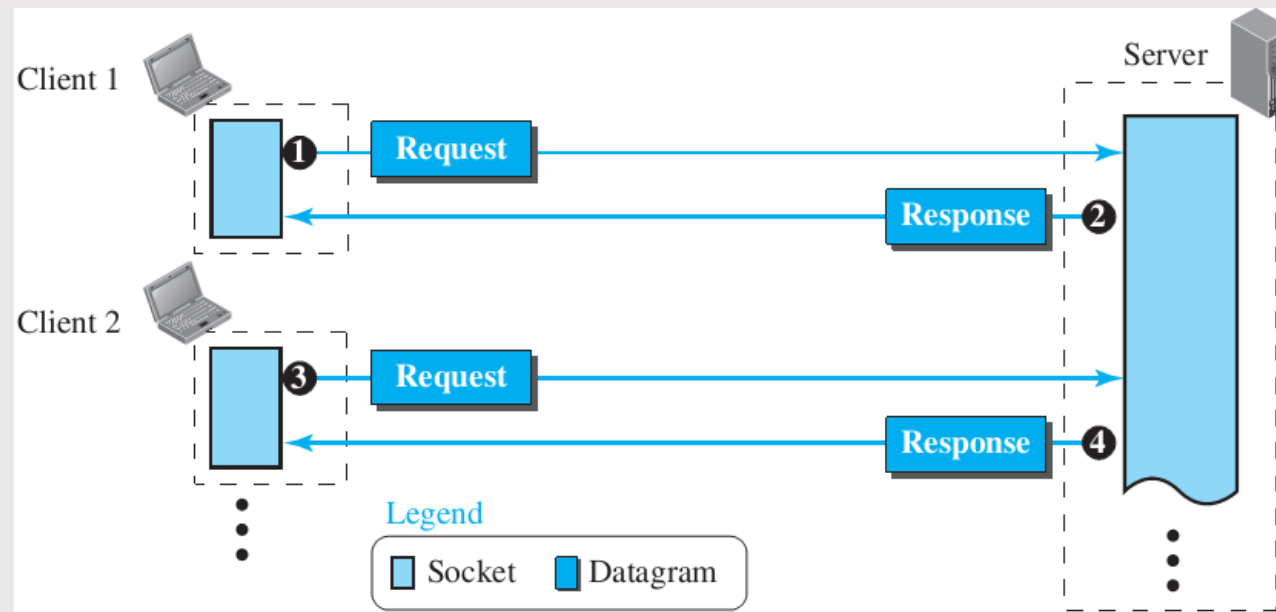
# REQUERIMIENTOS DE SERVICIO DE TRANSPORTE DE APLICACIONES COMUNES

Aplicación	Pérdidas	Ancho de banda	Sensible a retardo
Transferencia de archivos	No	Elastico	No
e-mail	No	Elástico	No
Documentos Web	No	Elástico	No
audio/video en tiempo real	Tolerante	Audio: 5Kbps-1Mbps Video: 10Kbps-5Mbps	Si, 100's ms
audio/video Imacenado	Tolerante	Igual al de arriba	Si, pocos se
Juegos interactivos	Tolerante	Pocos Kbps a mas	Si, 100's ms
Mensajería instantánea	No	Elástico	Si y No

# SERVICIOS DE LOS PROTOCOLOS DE TRANSPORTE EN INTERNET

## Servicio UDP:

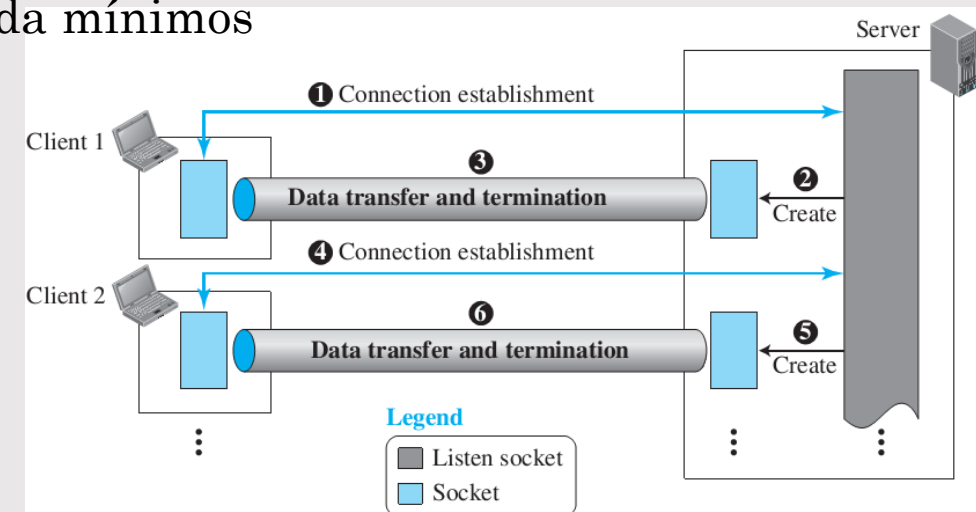
- Transferencia de datos no confiable entre proceso Tx y Rx.
- No provee: acuerdo entre los procesos, confiabilidad, control de flujo, control de congestión, garantías de retardo o ancho de banda



# SERVICIOS DE LOS PROTOCOLOS DE TRANSPORTE EN INTERNET

## Servicio TCP:

- *Orientado a la conexión* acuerdo requerido entre procesos cliente y servidor antes de transferencia
- *Transporte confiable* entre proceso Tx y Rx
- *Control de flujo*: Tx no sobrecargará al Rx
- *Control de congestión*: frena al Tx cuando la red está sobrecargada
- *No provee*: garantías de retardo ni ancho de banda mínimos



# APLICACIONES INTERNET

APLICACIÓN	PROTOCOLO DE CAPA DE APLICACIÓN	PROTOCOLO DE TRANSPORTE
Correo electrónico	SMTP (RFC 2821)	TCP
Remote Terminal Access	Telnet (RFC 854)	TCP
Web	HTTP (RFC 2616)	TCP
Transferencia de archivos	FTP (RFC 959)	TCP
Streaming	HTTP, RTP, Propietario	TCP/UDP
Telefonía IP	SIP, RTP, Propietario	UDP