Redes de Computadoras

Vicente González Ruiz
Depto de Arquitectura de Computadores y Electrónica
vruiz@ual.es
http://www.ace.ual.es/~vruiz/docencia

3 de junio de 2009

Contenidos

Parte I

Introducción a las redes de computadoras

Capítulo 1

¿Qué es Internet?

1.1. ¿Qué es Internet?

Existen muchas definiciones de Internet. Algunas de ellas:

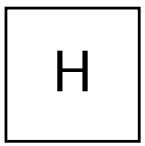
1. Internet es una red de millones de computadoras que se extiende a nivel mundial.
2. Internet es una estructura computacional en la que se ejecutan aplicaciones distribuidas (que intercambian datos a través de la red).
3. Internet es una gran internet (una red de redes).

1.2. ¿Qué es una internet?

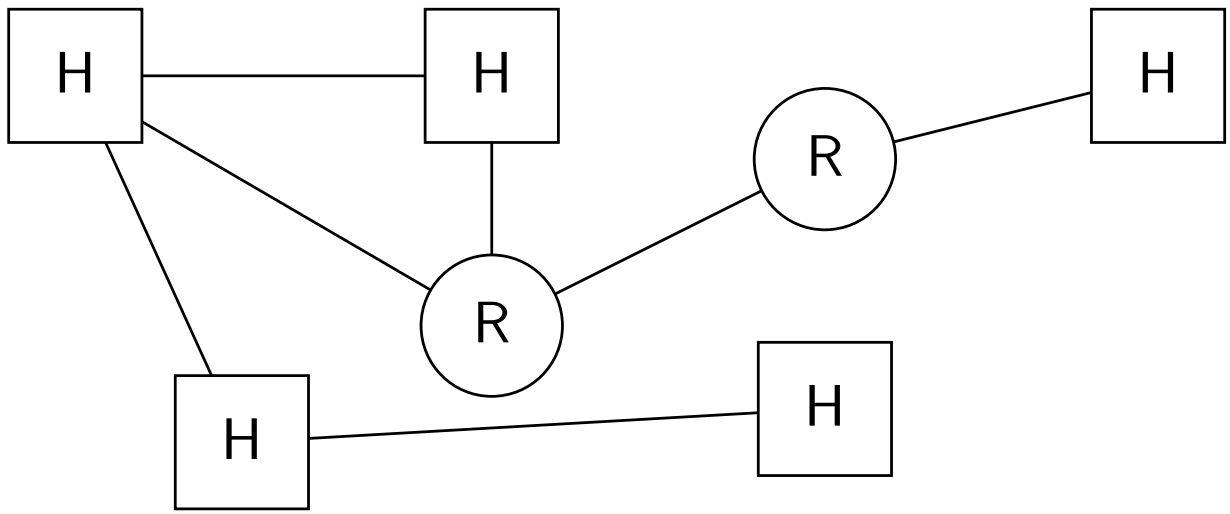
- Por definición, una internet es la estuctura formada cuando interconectamos dos (o más) redes. En otras palabras, *un conjunto de nodos de computación interconectados mediante enlaces de transmisión de datos* [14].
- Al nivel "internet", sólo existen dos tipos distintos de nodos:
  1. Hosts o sistemas terminales (end systems).
  2. Routers o dispositivos de conmutación de paquetes de datos (data packet switches).

1.3. ¿Que es un host?

- En Internet, un host es una computadora que ejecuta aplicaciones de red (que se comunica con otras aplicaciones en otros hosts).
- En adelante, generalmente denotaremos a un host mediante la figura:

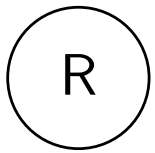


- Un host puede conectarse directamente a otros hosts o a otros routers.

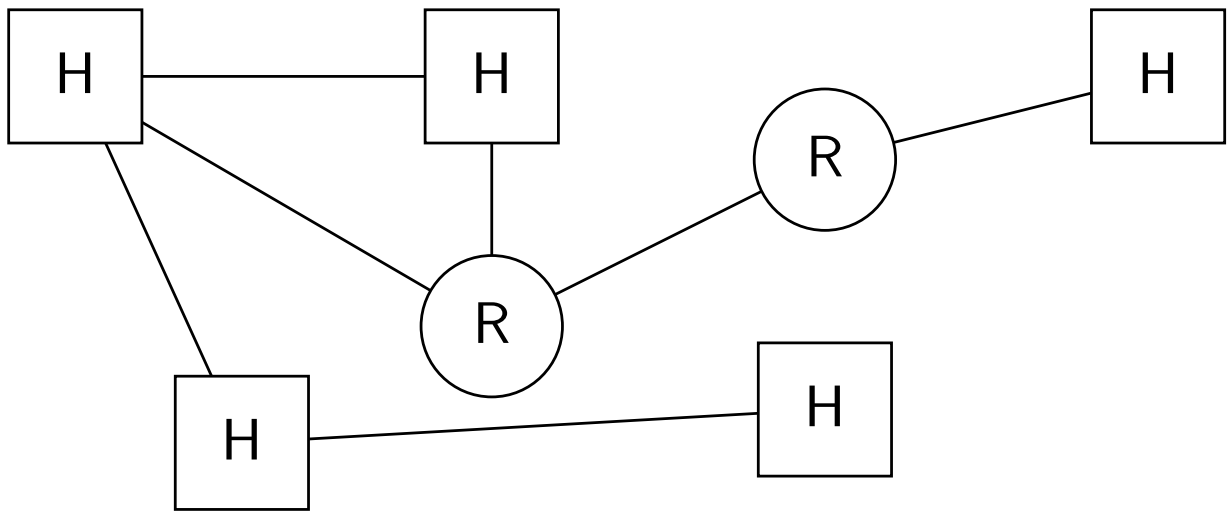


1.4. ¿Qué es un router?

- En Internet, un router es un sistema específicamente diseñado para intercambiar paquetes de datos con otros routers y hosts.
- En adelante, denotaremos un router mediante la figura:

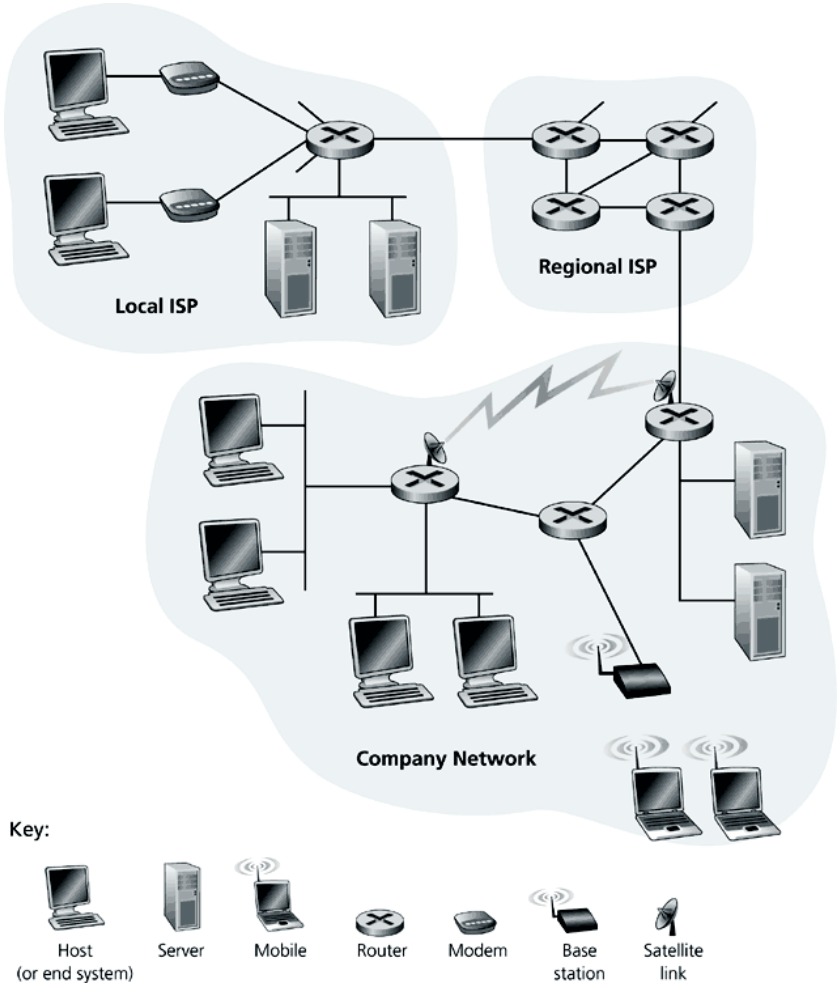


- Un router puede conectarse directamente a otros routers y/o hosts.

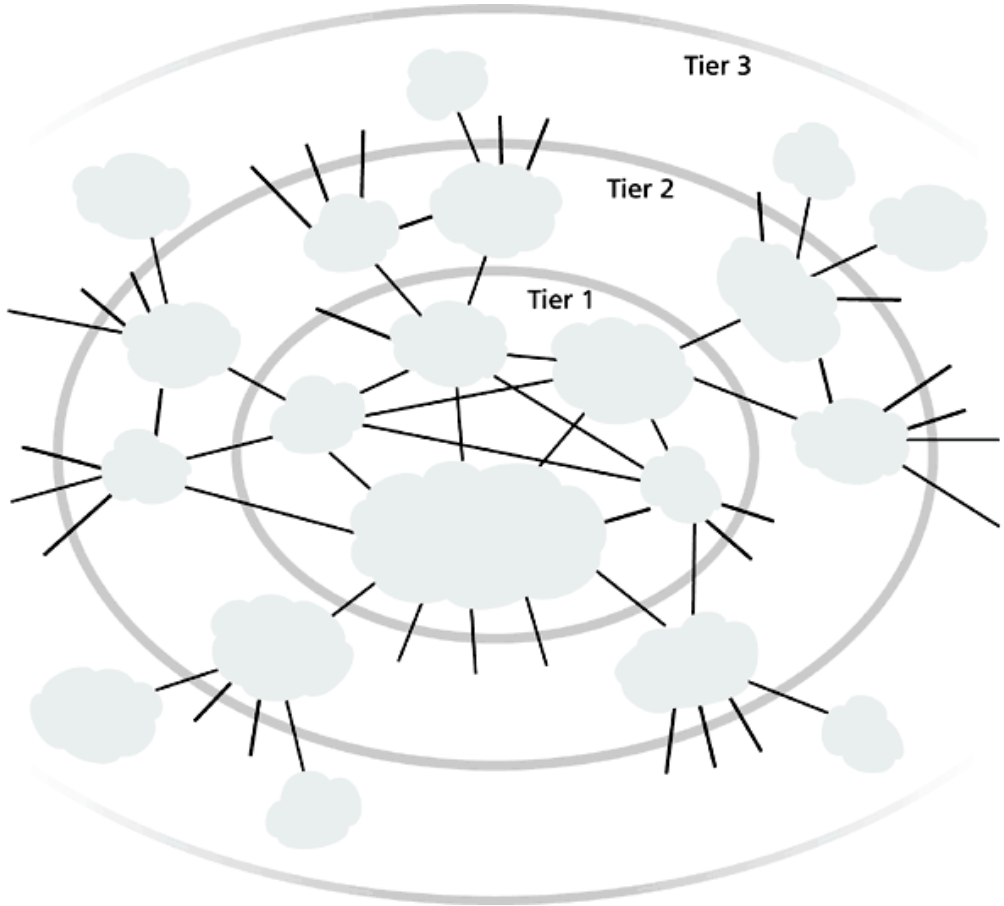


1.5. ¿Cuál es la morfología de Internet?

- Internet se organiza físicamente como una red de grandes redes que a su vez se componen de otras redes de tamaño menor.



- La situación geográfica de las redes se corresponde normalmente con los núcleos urbanos. Por tanto, si fuera visible desde el espacio, *Internet se parecería a una tela de araña (aunque mucho más caótica) compuesta por enlaces de transmisión de datos de larga distancia que interconectan internets más pequeñas*.



1.6. ¿Qué es un ISP?

- En la actualidad, la Internet pública es gestionada por una serie de empresas que dan servicio de conexión a Internet. A esta clase de empresas se les llama Internet Service Providers (proveedores de servicio de Internet) o ISP's.
- Existen ISP's que interconectan redes a nivel mundial y otros que lo hacen a escala menor. Dependiendo de esto hablaremos de ISP's de nivel 1 (nivel mundial), ISP's de nivel 2 (nivel nacional), etc., hasta llegar a los ISP's que dan servicio a las universidades, empresas, casas, etc.
- Un aspecto interesante e importante de cara al funcionamiento de Internet es que cada ISP se gestiona de forma independiente.

1.7. ¿Qué servicios proporciona la red?

- **A nivel de usuario:**
  1. Web surfing.
  2. Mensajería instantánea.
  3. Acceso remoto (remote login).
  4. Correo electrónico (e-mail).
  5. Compartición de archivos peer-to-peer (par-a-par) o P2P.
  6. Telefonía IP (Internet Protocol).
  7. Streaming de audio y vídeo.
  8. Juegos en red.
  9. Etc.
- **A nivel de aplicación:** existe un conjunto de bibliotecas de software que permiten transmitir datos entre las apliaciones distribuidas (anteriormente mencionadas) que corren en los hosts.

1.8. ¿Cómo usan los servicios las aplicaciones?

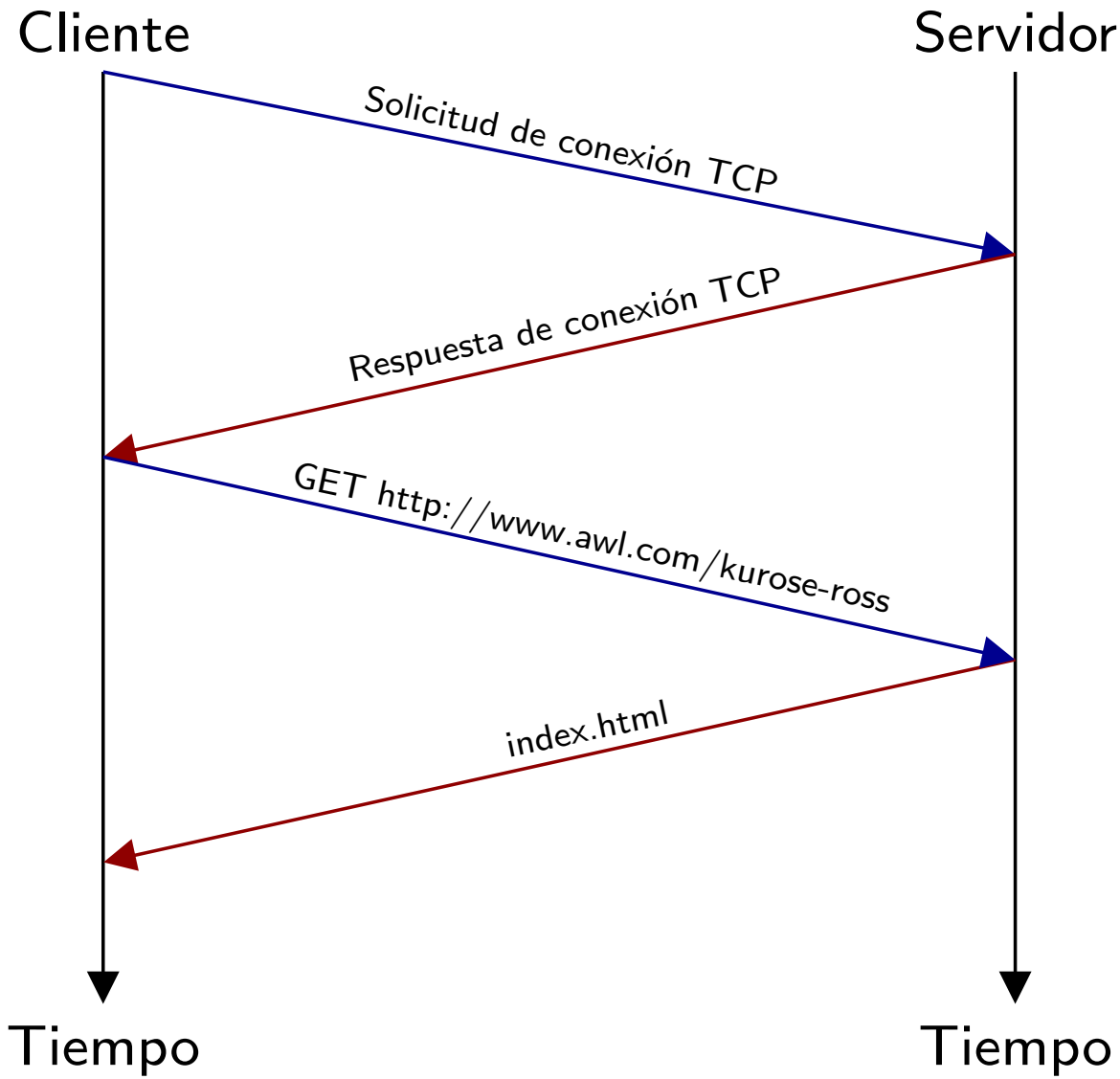
- Cualquier aplicación, que hace uso de Internet, debe usar estos tipos de servicios (uno o ambos a la vez):
  - **Servicios fiables orientados a conexión**, que garantizan que los datos son siempre entregados correctamente a la apliación receptora.
  - **Servicios no fiables y sin conexión**, que no garantizan nada. Únicamente que se intentará entregar los paquetes de datos.
- No existen restricciones temporales para realizar dichos servicios.

1.9. ¿Qué es un protocolo de red?

- En Internet, los servicios de los que hemos hablado se presentan a las aplicaciones en forma de protocolos de red.
- En general, *un protocolo es un conjunto de reglas de comportamiento que permiten a dos o más aplicaciones remotas comunicarse entre sí*.
- En Internet, los protocolos más famosos son el TCP (Transmision Control Protocol) y el IP (Internet Protocol). El primero implementa el servicio fiable orientado a conexión y el segundo se encarga del routing (conducción) de los paquetes de datos a través de Internet.

1.10. ¿Qué son los clientes y los servidores?

- La mayoría de las aplicaciones distribuidas siguen el paradigma cliente/servidor. En él, *la aplicación cliente es la que inicia la comunicación y la aplicación servidora contesta a las peticiones*.
- Por ejemplo, cuando navegamos a través de la Web, utilizamos el TCP para comunicarnos con el servidor Web. A continuación se muestra lo que ocurre cuando descargamos una página Web:



En este ejemplo, nuestro navegador Web es la aplicación cliente y el servidor Web es la aplicación servidora.

1.11. ¿Qué es un servicio orientado a conexión?

1. Es el servicio proporcionado por la red que simula una conexión exclusiva entre dos aplicaciones de red.

2. La aplicación receptora recibe una copia exácta del flujo de bytes generado por la aplicación emisora. Para ello, antes del comienzo de la transmisión, el emisor se asegura de que el receptor puede atender la llegada de los datos.

3. Implementa un mecanismo de control de flujo y de congestión. Mediante el primero, un emisor rápido no colapsa a un receptor más lento que él. Gracias al segundo, un emisor disminuye su tasa de transmisión cuando la red se colapsa por exceso de trabajo.

4. Aunque garantiza la entrega correcta de los datos, no se especifica un tiempo máximo. Los datos enviados se agrupan en paquetes para transmitirse, y estos pueden sufir un retraso indeterminado en los routers.

5. Lo proporciona el TCP que está definido en el RFC (Request For Comments) 793 [5].

1.12. ¿Qué es un servicio no confiable y sin conexión?

1. La aplicación emisora envía los datos a la aplicación receptora sin necesidad de conocer si ésta los está esperando. No existe establecimiento de conexión.

2. Los datos se envían agrupados en paquetes, que se pueden retrasar un tiempo indeterminado, perder, modificar y desordenar.

3. La red no proporciona ninguna información acerca de qué les ha pasado a los paquetes.

4. La aplicación emisora envía los datos a la velocidad que le parece (normalmente tan rápido como puede): no existen mecanismos de control de flujo y de congestión.

5. Lo proporciona el UDP (User Datagram Protocol) que se define en el RFC 768 [23].

Capítulo 2

Técnicas de transmisión de datos

2.1. Conmutación de circuitos

- Actualmente se utiliza en telefonía [32] porque garantiza una tasa de transmisión de bits (bit-rate) constante entre el emisor y el (o los) receptor(es).
- Los canales en los enlaces de transmisión y conmutadores (switches), es decir, los "circuitos" quedan reservados durante el tiempo que dura la comunicación.[1]
- Una vez que se ha establecido la conexión, no existe tiempo de espera para transmitir. Sin embargo, la conexión puede denegarse si no existen suficientes circuitos (red congestionada).
- Los recursos siguen asignados aunque los emisores dejen temporalmente de transmitir. Bajo esta circunstancia, los circuitos se desperdician.

[1] En el fondo es como si siguiera existiendo la operadora que debe enchufar algunas clavijas para establecer la conexión.

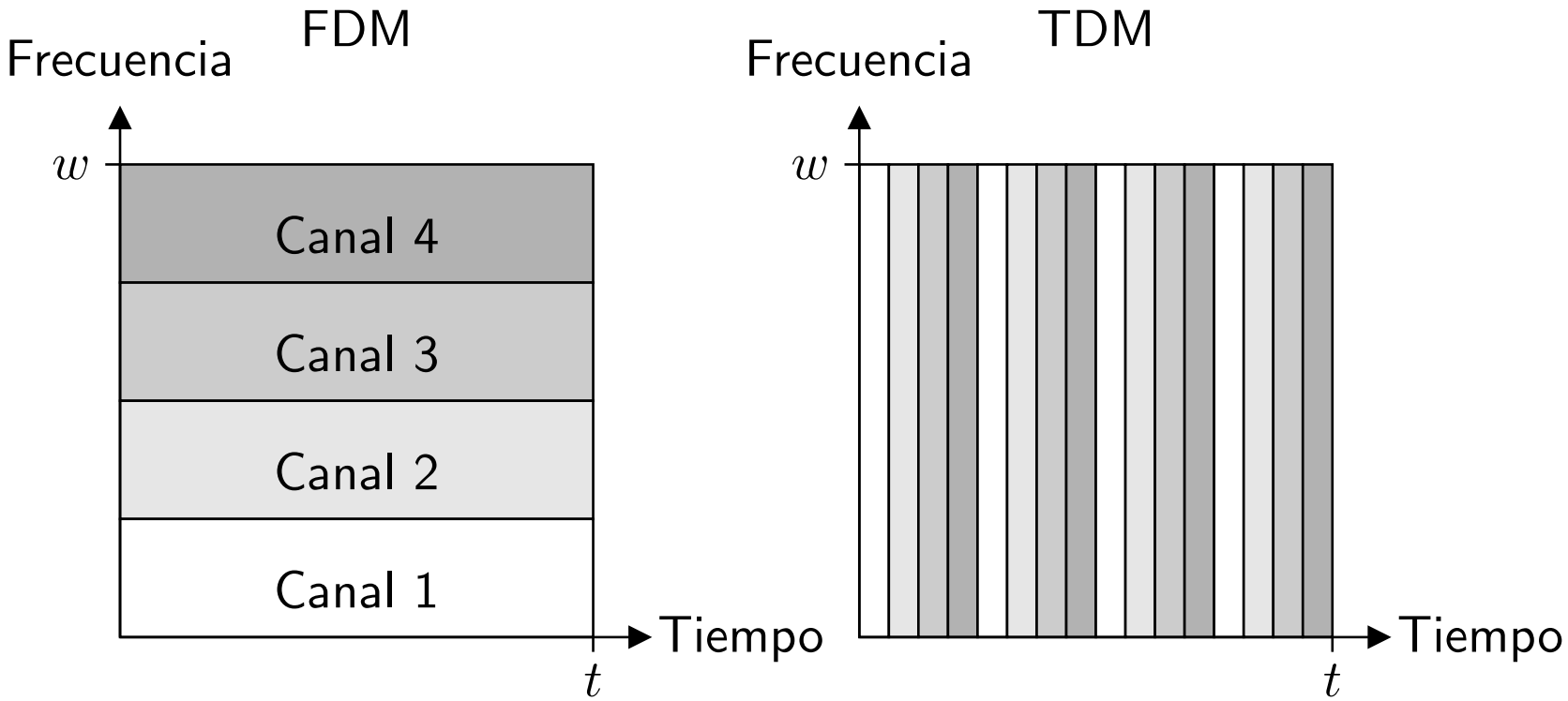
2.2. Conmutación de paquetes

- La red no reserva recursos de transmisión en concreto, a lo sumo establece prioridades.
- La tasa de transmisión es variable y depende de la carga de la red, con o sin establecimiento de conexión.

2.3. Multiplexación en redes de conmutación de circuitos

- Normalmente los enlaces presentan una capacidad mayor de la que se necesita en una transmisión. Por este motivo se multiplexan muchas transmisiones mediante dos técnicas distintas, aunque equivalentes desde el punto de vista de la tasa de bits proporcionada:

  1. **Multiplexación en la frecuencia o FDM (Frecuency Division Multiplexing):** Los circuitos se implementan mediante canales de frecuencia [27] (véase el Apéndice C).
  2. **Multiplexación en el tiempo o TDM (Time Division Multiplexing):** Los circuitos se implementan mediante slots de tiempo periódicamente asignados.

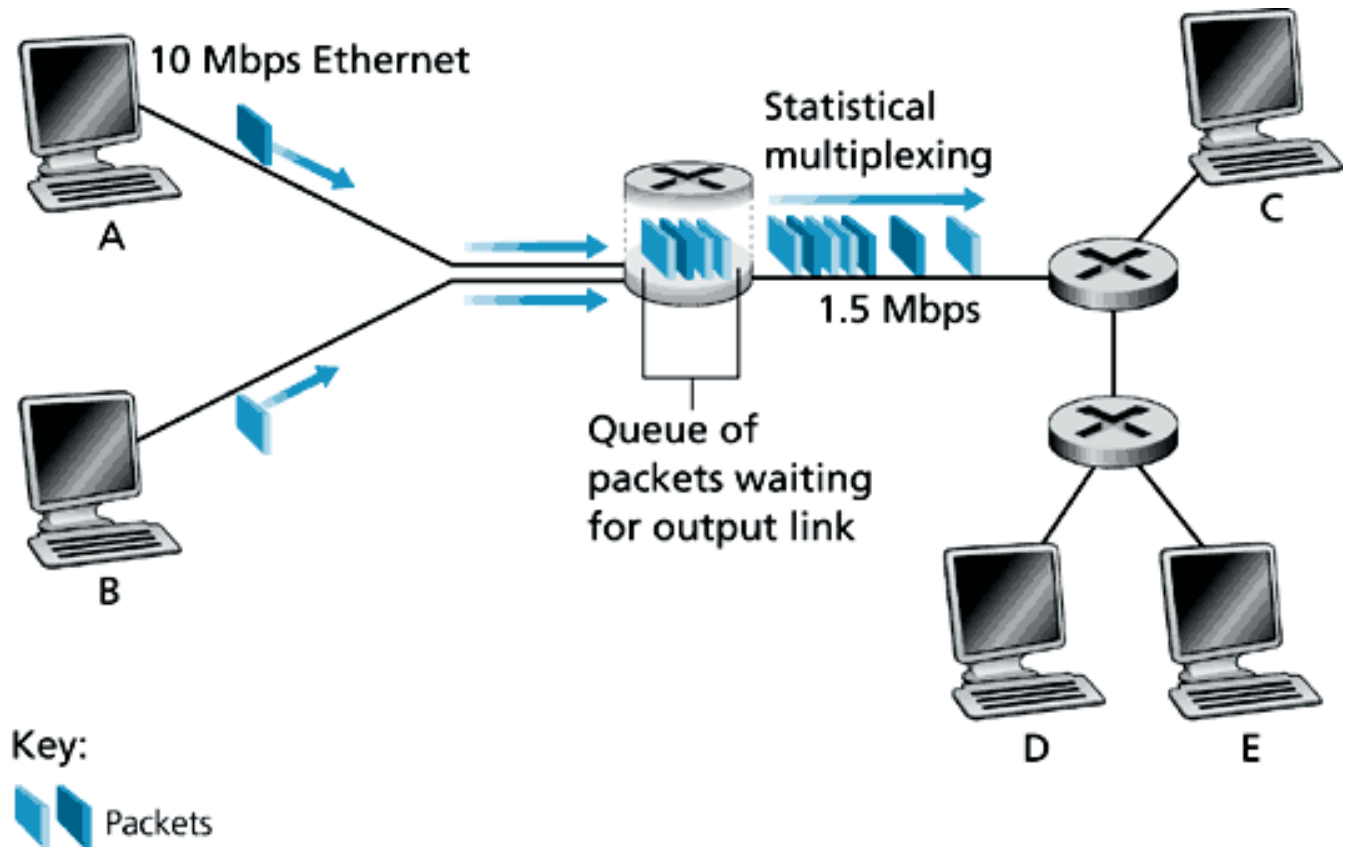


Nótese que la capacidad de transmisión de cada emisor es la misma en ambos casos e igual a:

$$\frac{w}{4} \times t$$

2.4. Multiplexación en redes de conmutación de paquetes

- Es una generalización de TDM, donde los slots de tiempo suelen ser mucho más grandes y de una longitud variable en función del tamaño del paquete.



- Los slots de tiempo no quedan reservados. Esto provoca demoras impredecibles en los conmutadores de paquetes dependiendo de la carga.
- En promedio (olvidando los overheads provocados por las cabeceras) la capacidad de transmisión asignada a los emisores es la misma que en conmutación de circuitos.

2.5. ¿Por qué Internet utiliza conmutación de paquetes?

- **Mayor aprovechamiento de los recursos de la red [14]:** El ancho de banda disponible se aprovecha mejor porque las aplicaciones suelen transmitir de forma aleatoria e intermitente (sin un patrón predeterminado). En otras palabras: se transmite un volumen total de datos mayor.

- **Posibilidad de tasas de transmisión instantáneas muy altas**: La red en general tiene mucha capacidad y si da la casualidad de que la usamos "solos", utilizaremos todo el ancho de banda disposible.

2.6. ¿Qué inconvenientes genera la conmutación de paquetes?

- **Tasa de transmisión variable**: El bit-rate depende de la carga.
- **Jitter**[2] $>$ **0**: El retraso que sufren los paquetes es variable.

[2] La variación (en términos absolutos, es decir, sin considerar el signo) de la latencia de la red.

2.7. Almacenamiento y reenvío

- Los conmutadores de paquetes en Internet normalmente utilizan la técnica de esperar a recibir completamente un paquete antes de proceder a su retransmisión (store-and-forward). Motivos:

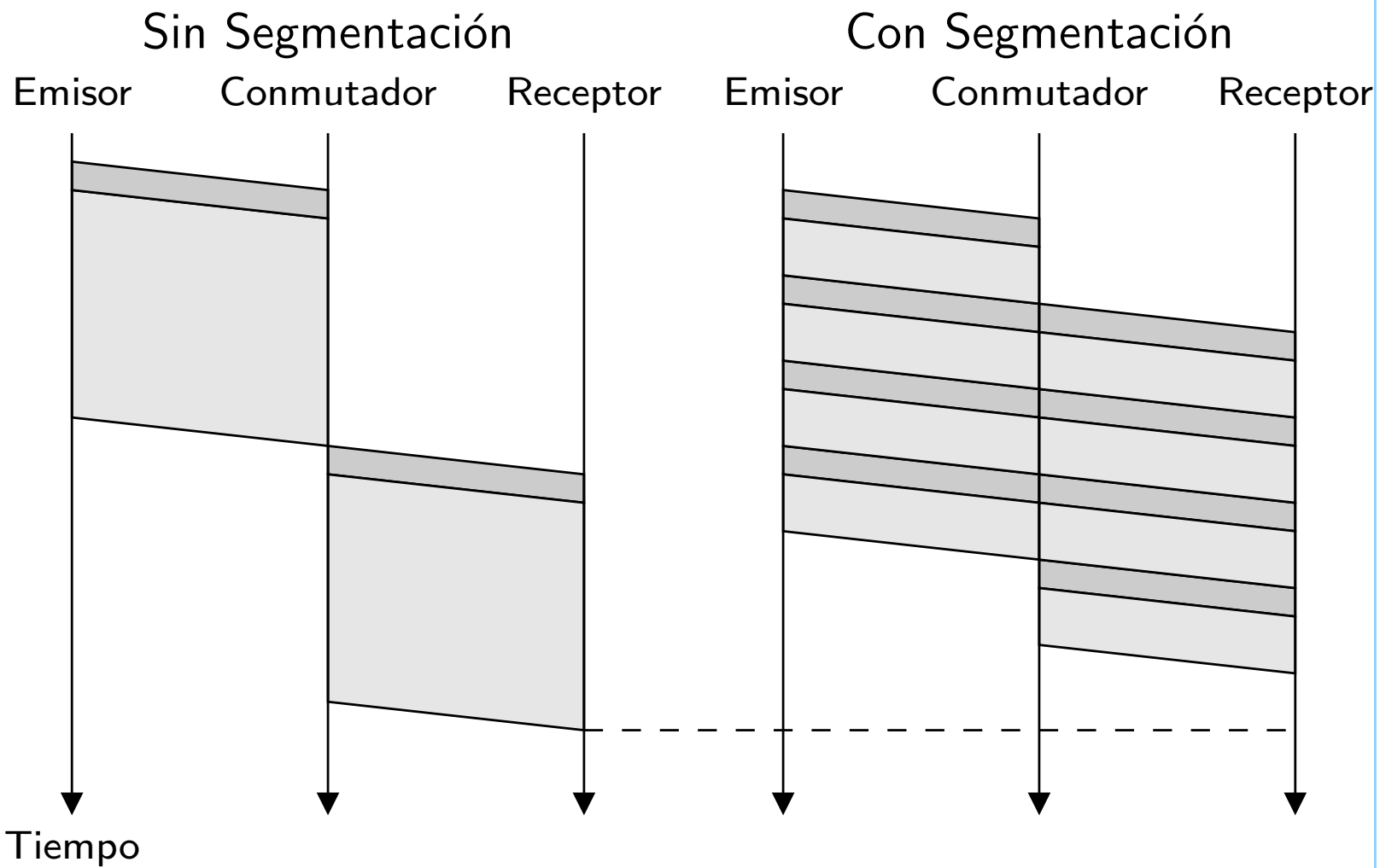
  1. El enlace de salida puede estar ocupado transmitiendo algún paquete anterior.
  2. La tasa de transmisión del enlace de entrada y de salida puede ser distinta.

- La alternativa (cut-through) es muy rara y sólo se utiliza cuando estos dos factores no son un problema (por ejemplo, en máquinas multiprocesadoras).

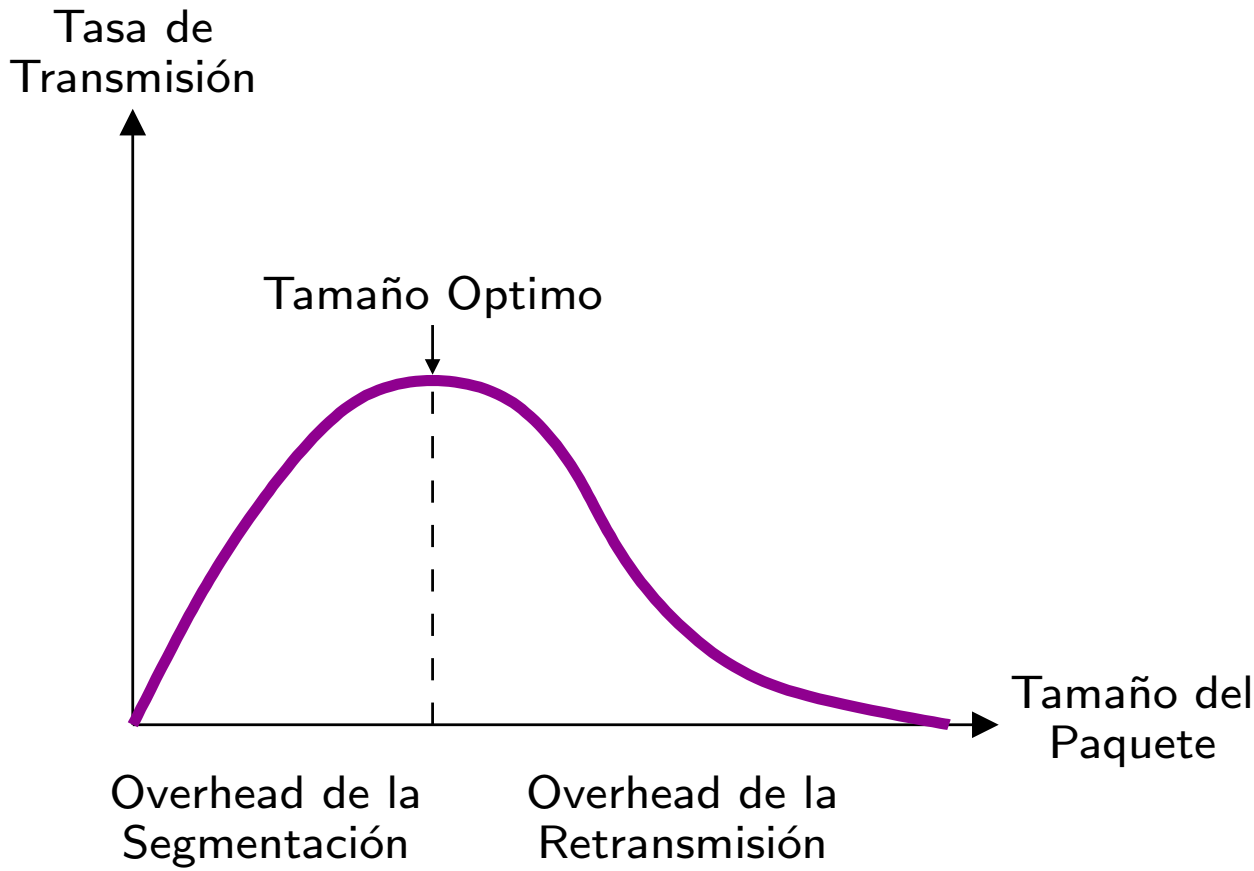
2.8. Segmentación de los mensajes

- Los mensajes (de longitud arbitraria) son divididos (normalmente por el emisor) en paquetes de datos antes de ser enviados. El receptor se encarga de reensamblarlos. Esto se hace fundamentalmente por los siguientes motivos:

1. Aunque la segmentación introduce redundancia (en forma de cabeceras), los tiempos de transmisión de los mensajes en redes conmutadas son menores cuando utilizamos almacenamiento y reenvío, ya que disminuye la latencia [27].



2. La retransmisión de los paquetes erróneos (la forma más frecuente de corrección de errores) introduce menos redundancia [32] ya que los paquetes son más pequeños.

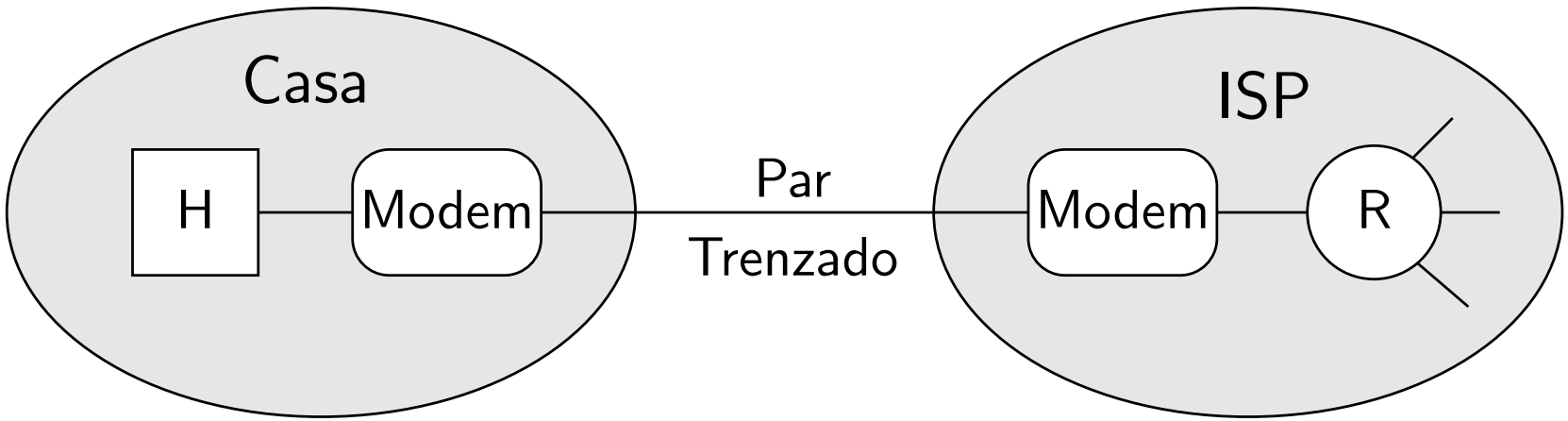


Capítulo 3

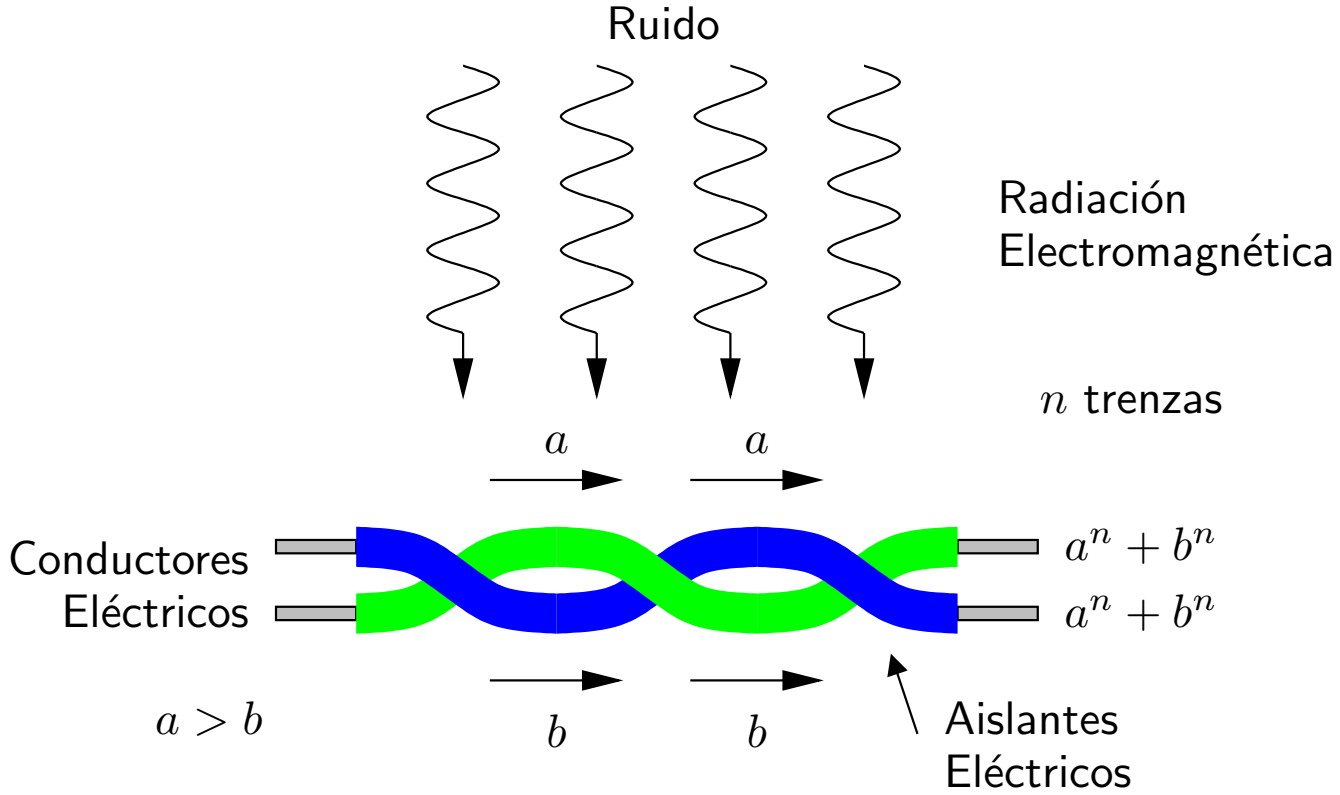
Técnologías de trasmisión de datos

3.1. Teléfono de voz

- Se utiliza fundamentalmente en acceso residencial y hasta hace poco ha sido (con mucho) el sistema de acceso a Internet más popular debido a su menor coste.
- Se utiliza un canal de voz (con 4 KHz de ancho de banda) para transmitir, que permite a lo sumo 56 Kbps de tasa de transmisión (normalmente menos debido al ruido en la línea) mediante modulación QAM (modulación en amplitud y fase simultáneas, véase el Apéndice C).
- Son conexiones punto a punto entre el host y el ISP [14].



- El enlace se implementa mediante un par de hilos trenzados de cobre. Las trenzas contribuyen a tener el mismo ruido en ambos hilos. De hecho, los enlaces se clasifican en categorías dependiendo del número de trenzas que poseen por unidad de longitud.[1]



- La distancia entre los modems puede ser muy alta (cientos de Km's).

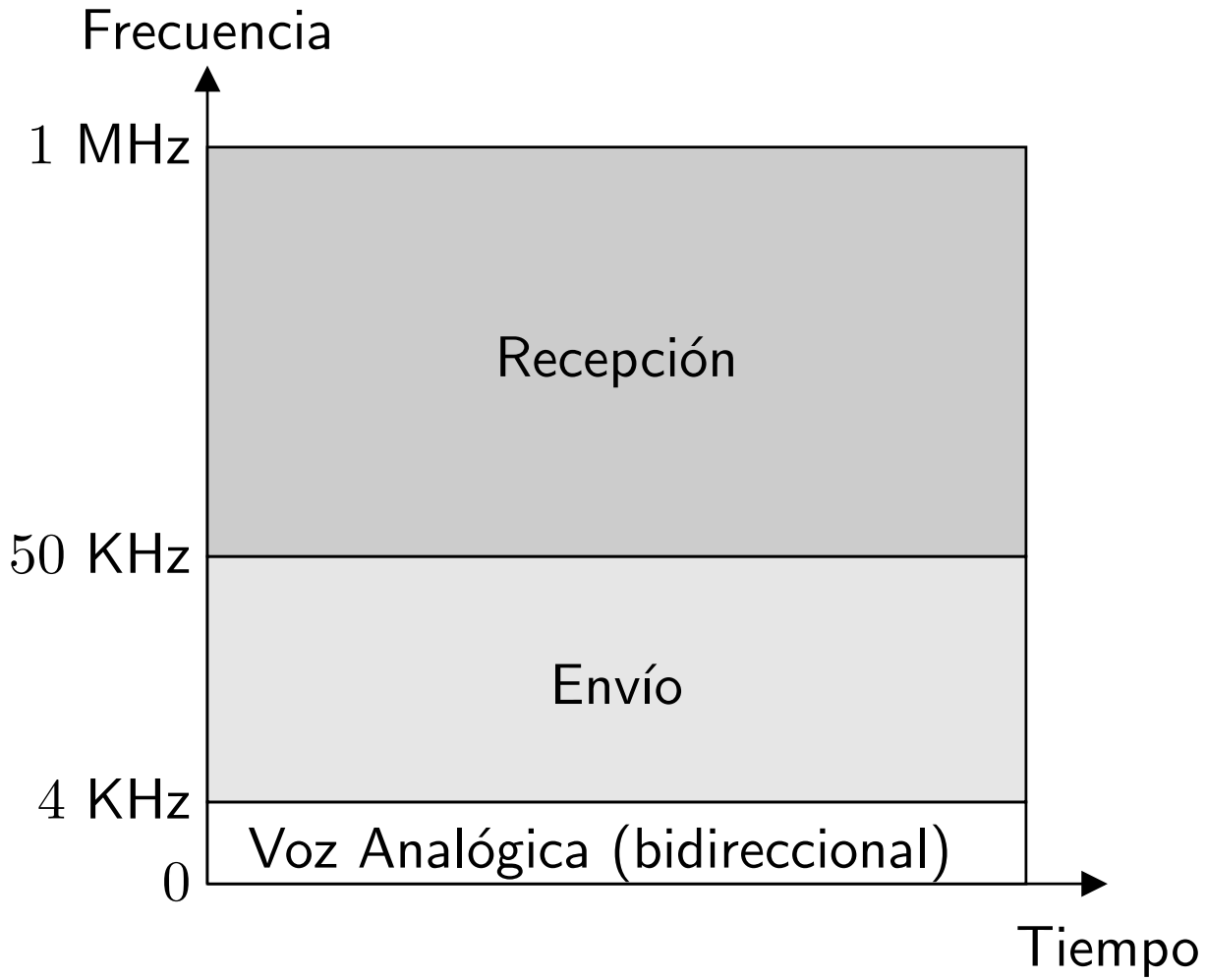
[1] Por ejemplo, en redes es muy común usar UTP (Unshield Twisted Pair) categoría 5 con los que alcanzan tasas de 100 Mbps sobre unas decenas de metros.

3.2. ADSL (Asymetric Digital Subscriber Line)

- Se utiliza fundamentalmente en acceso residencial.
- Emplea la misma infraestructura que el teléfono de voz (par trenzado).
- Permite alcanzar hasta 20 Mbps (dependiendo de la distancia entre los modems)[2] y usar el teléfono simultáneamente [27].
- Los enlaces ADSL son punto a punto y dedicados (no se comparten con otros vecinos).

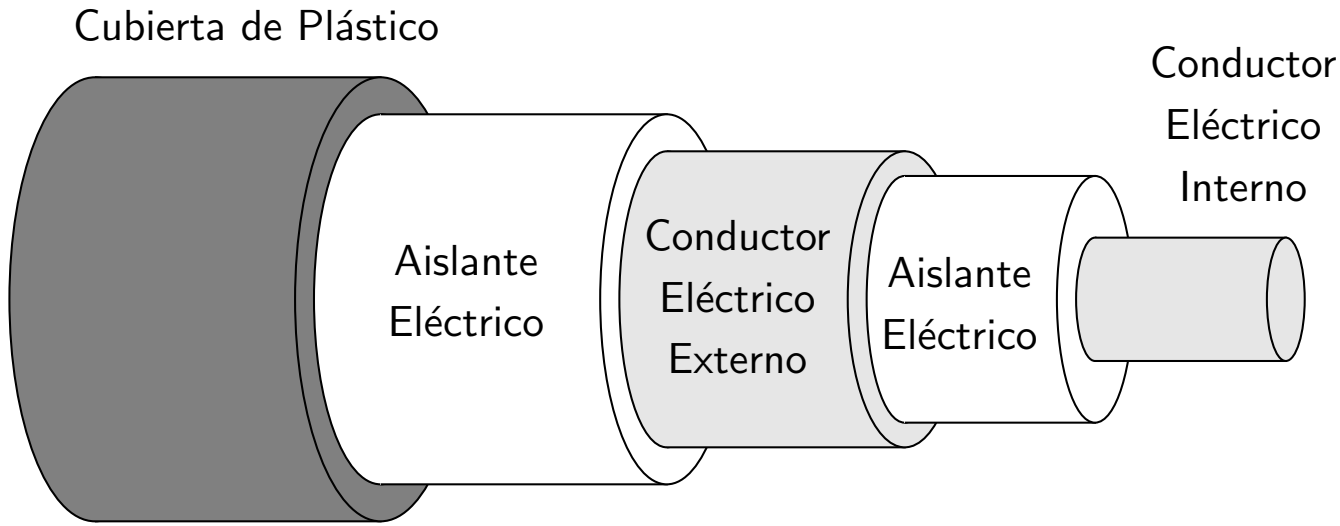
[2] Para alcanzar estas velocidades generalmente sólo puede haber unas decenas de metros entre ambos modems.

- Por la forma en que los usuarios suelen usar Internet, el canal de bajada generalmente tiene un ancho de banda mayor que el de subida (de ahí lo de asimétrico). Se utiliza multiplexación en frecuencia.

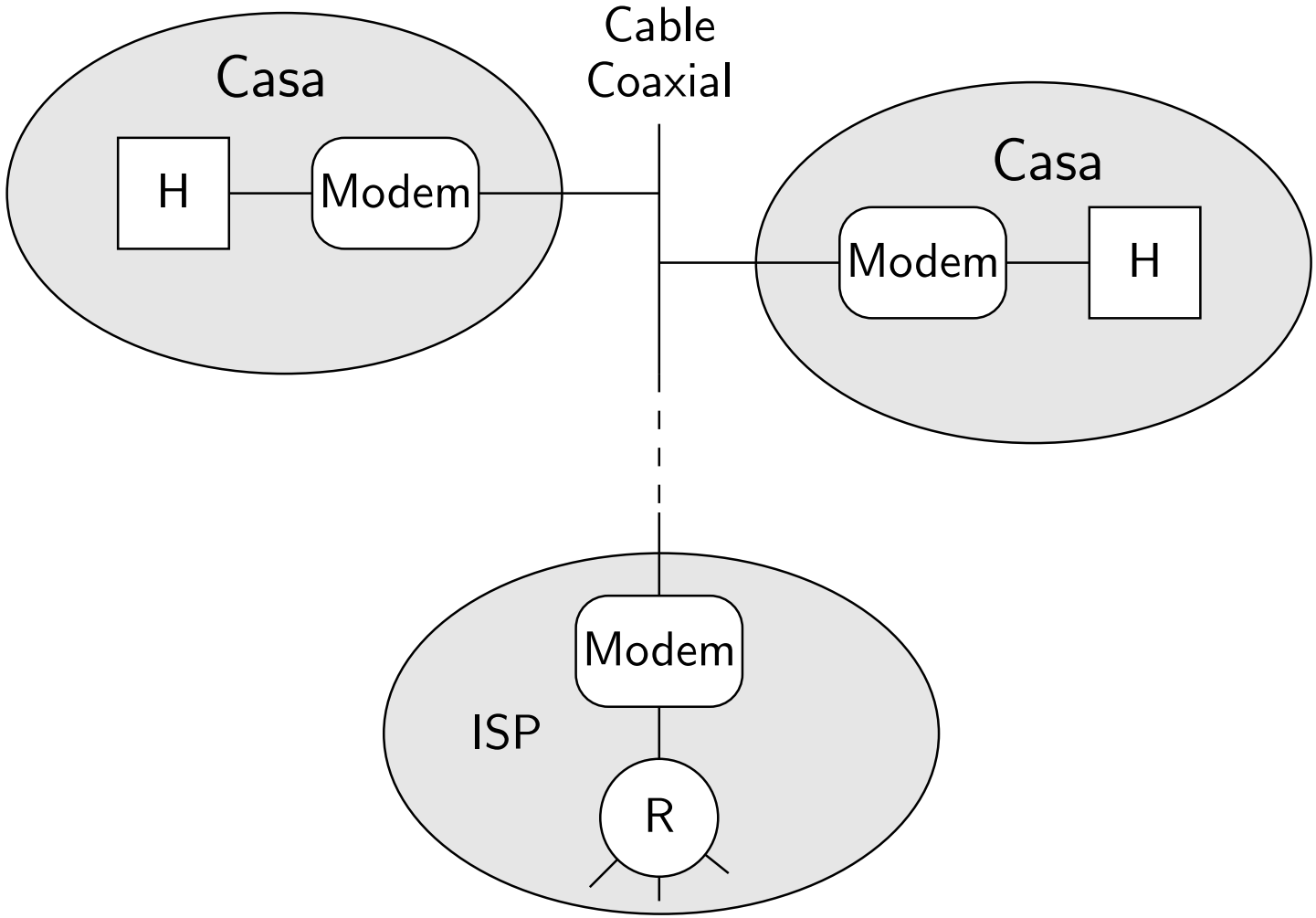


3.3. Cable coaxial de TV

- Se utiliza fundamentalmente en acceso residencial.
- Aprovecha la infraestructura de la TV por cable. La idea es dedicar algunos canales de TV (unos 6 MHz/canal) para enviar datos y otros para recibir.
- El medio de transmisión (cable coaxial) es compartido por todos los usuarios de Internet del vecindario. A su vez los canales pueden compartirse dependiendo del número de usuarios.

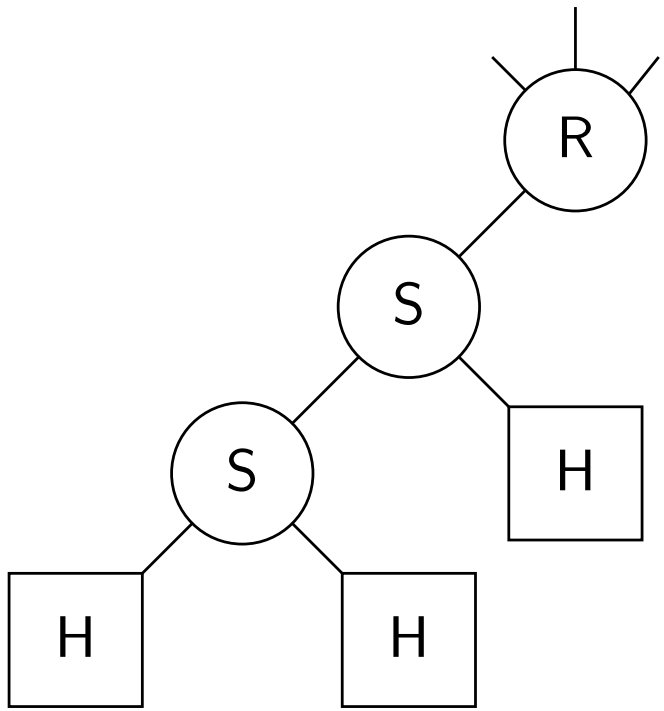


Si esto ocurre, el canal de carga (subida) presenta problemas de colisiones y el de descarga (bajada) es compartido por todos los vecinos. Estos problemas no suelen (o al menos no deberían) bajar la tasa de bits obtenida por debajo de la contratada.



3.4. Ethernet conmutada

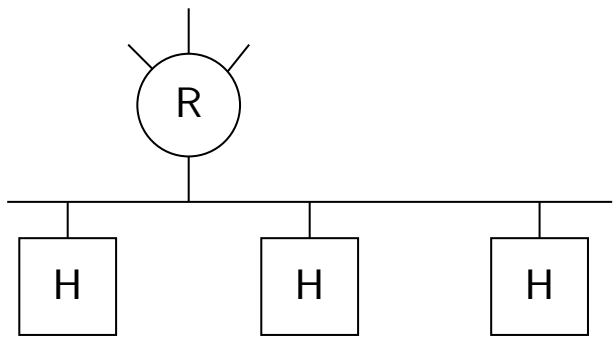
- Es la tecnología LAN (Local Area Network) más implantada en empresas, universidades, etc.
- Los hosts se conectan mediante enlaces punto a punto a un conmutador de tramas Ethernet, formándose típicamente estructuras en árbol.



- Utiliza enlaces de par trenzado (distancias cortas) o fibra óptica (distancias largas).
- Las tasas de transmisión típicas son 100 Mbps y 1 Gbps entre cada par de nodos.
- No existen colisiones. El conmutador las resuelve.

3.5. Wireless LAN o Wi-Fi

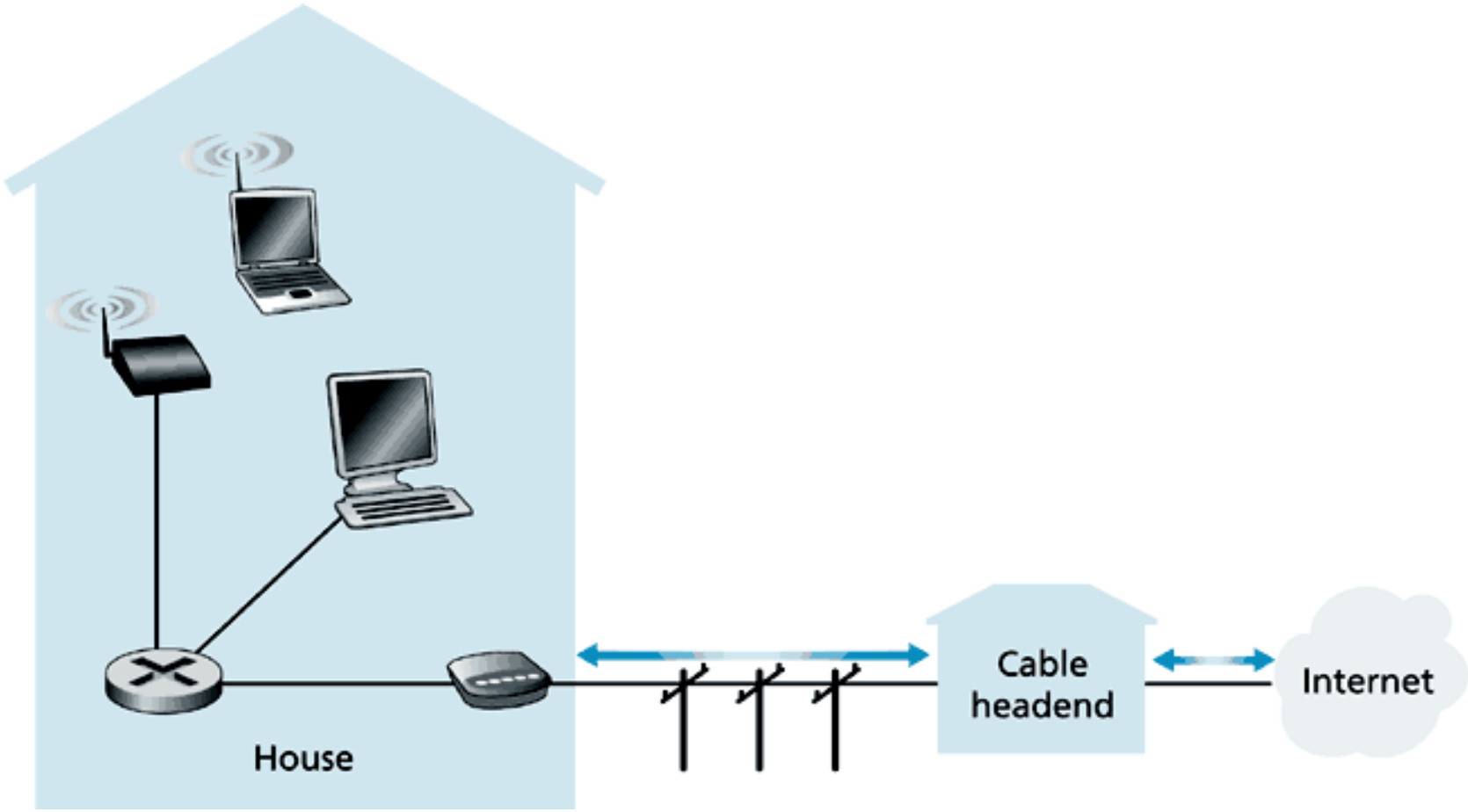
- Está reemplazando a la Ethernet conmutada cuando es necesaria cierta movilidad de los hosts (bibliotecas, cafeterías, universidades, casas).
- Permite alcanzar tasas de 54 Mbps (802.11g) o 11 Mbps (802.11b), aunque ésta puede decrecer al disminuir la SNR[3] (véase el Apéndice B).
- Se basa en la tecnología Ethernet para medios compartidos[4] (en este caso se utilizan enlaces de radio), donde una serie de hosts comparten en mismo rango de frecuencias para enviar y recibir (TDM).



[3] 5 Mbps, 2 Mbps, 1 Mbps ...

[4] Inicialmente las redes Ethernet usaban un cable coaxial para interconectar todos los hosts de la red.

- La distancia entre los nodos y los switches no sobrepasa normalmente las decenas de metros (dependiendo de las paredes, etc.). Un caso típico:



3.6. Telefonía móvil digital celular

- Permiten conexiones de hasta unas decenas de kilometros.
- Existen muchas normas. Las más usadas [2]:
  1. GSM (Global System for Mobile communications): 14 kbps/hasta 14 kbps (subida/bajada).
  2. GPRS (General Packet Radio Service): 14 kbps/hasta 64 kbps.
  3. 3G(GSM) (3rd Generation GSM): 348 kbps.

3.7. Telefonía móvil digital por satélite

- Permiten conectarnos desde cualquier punto del planeta gracias a una constelación de satélites [33].
- Los satélites son no geo-estacionarios con el objeto de que puedan estar a unos pocos cientos de Km de la superficie terrestre.[5]
- Los satélites no geo-estacionarios están constantemente en movimiento respecto de la superficie.[6]
- Se consiguen tasas de transmisión de hasta 64 Kbps.

[5] Los satélites estacionarios (que no se mueven respecto de la superficie terrestre) están generalmente demasiado lejos (a unos $36{,}000$ Km de la superficie de La Tierra) como para que con la potencia de transmisión que usan los móviles la conexión sea posible (la SNR de la transmisión sea suficientemente alta) (veáse el Apéndice B).

[6] Se mantienen a una distancia constante de la superficie porque la fuerza centrífuga es igual a la fuerza de la gravedad.

3.8. ATM (Asynchronous Transfer Mode)

- Es una tecnología desarrollada para implementar WANs (Wide Area Network) y está compuesta por conmutadores y enlaces punto a punto de larga distancia. No existe ninguna topología específica aunque la más frecuente suele ser en estrella.
- Se utiliza fundamentalmente en transmisiones de larga distancia entre ISPs.
- Los enlaces pueden alcanzar una gran variedad de velocidades, pero normalmente se manejan Gbps.
- Los enlaces se implementan generalmente mediante fibra óptica o microondas (tanto terrestres como a través de satélites[7]), aunque para distancias muy cortas puede usarse par trenzado.

[7] Principalmente geo-estacionarios.

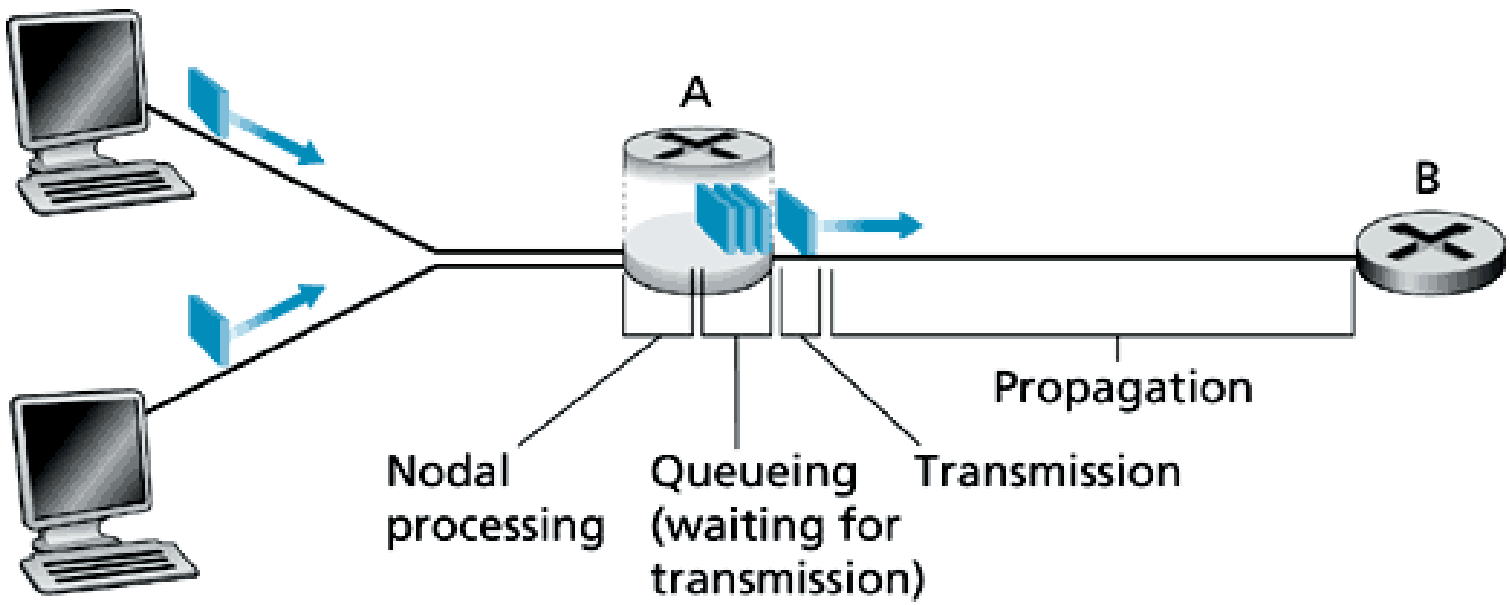
- Permite reservar ancho de banda (emulando la conmutación de circuitos muy bien) y por este motivo ATM es fundamentalmente utilizadas por las compañías telefónicas.

Capítulo 4

Retardos y pérdidas en redes de conmutación de paquetes

4.1. Funcionamiento básico de un router

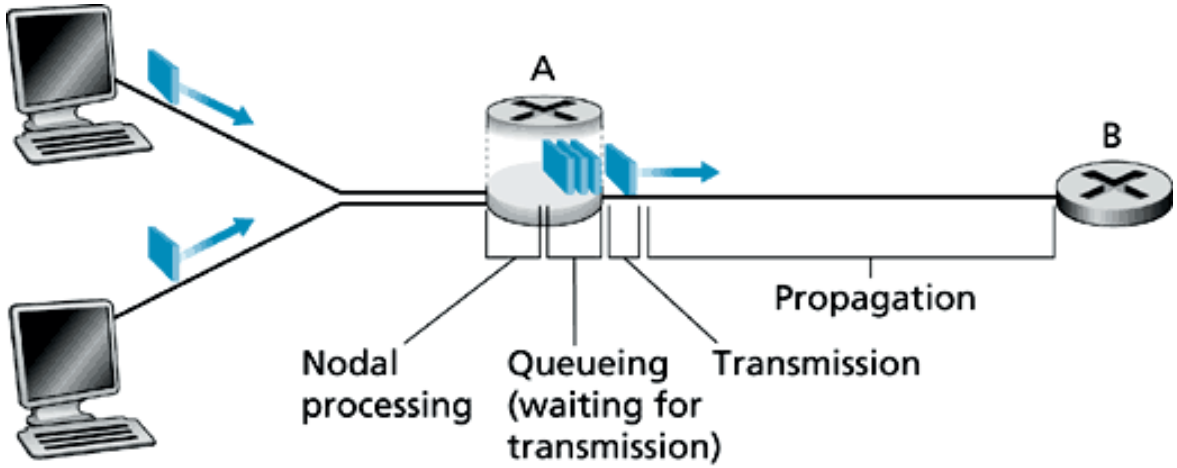
- Dispositivo de conmutación encargado de conducir los paquetes desde un emisor hasta el/los receptor/es. El tratamiento de cada paquete es independiente (el paquete es la máxima unidad de transmisión) [14].
- Asociado a cada enlace de salida existe una cola (memoria) donde esperan los paquetes que van a ser retransmitidos por ese enlace de salida.



- Aunque no exista ningún paquete almacenado en la cola de salida, la retransmisión de un paquete entrante no comienza hasta que se ha recibido totalmente (store-and-forward). Como alternativa, si el enlace está ocioso podría retransmitirse el paquete conforme se recibe (cut-through). Sin embargo, esta técnica es muy rara.
- Las colas de salida son finitas y cuando se llenan de paquetes, los nuevos paquetes entrantes se descartan (son destruidos).
- Normalmente los paquetes son colocados en la cola por estricto orden de llegada (no existen prioridades).

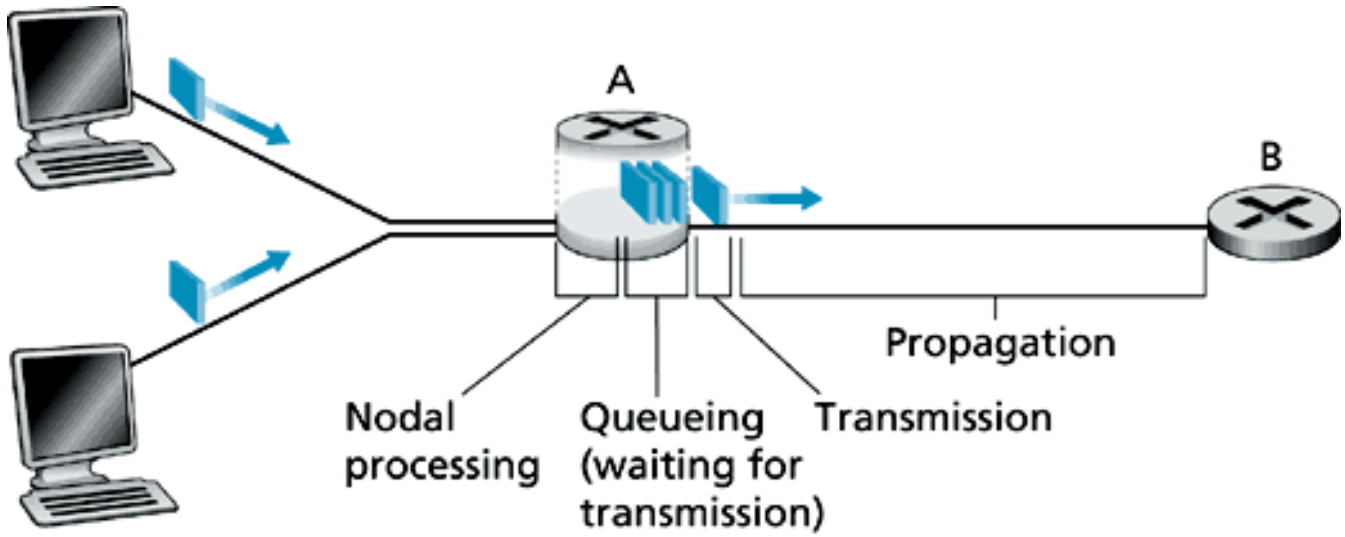
4.2. Tiempo de procesamiento (t_{proc})

- Tiempo necesario para que un router examine la cabecera de un paquete y determine hacia qué enlace de salida debe encaminarse.
- Incluye el tiempo necesario para determinar si existen errores de transmisión (alteraciones de bits) en la cabecera (no en los datos). Si existen, el paquete se desecha pues probablemente se entregaría a un receptor equivocado o el receptor no podría recuperar adecuadamente el paquete.
- El tiempo de procesamiento suele ser constante y del orden de los microsegundos.


A
B
Propagation
Nodal processing
Queueing (waiting for transmission)
Transmission

4.3. Tiempo de cola (t_{cola})

- Tiempo que un paquete espera en la cola de salida de un router a ser retransmitido.



- Depende del tiempo que necesiten los paquetes anteriores almacenados en la cola en ser transmitidos.
- Es 0 si la cola está vacía.

- Suele ser muy variable pues depende de la carga. Se mide en milisegundos. Véase el applet:

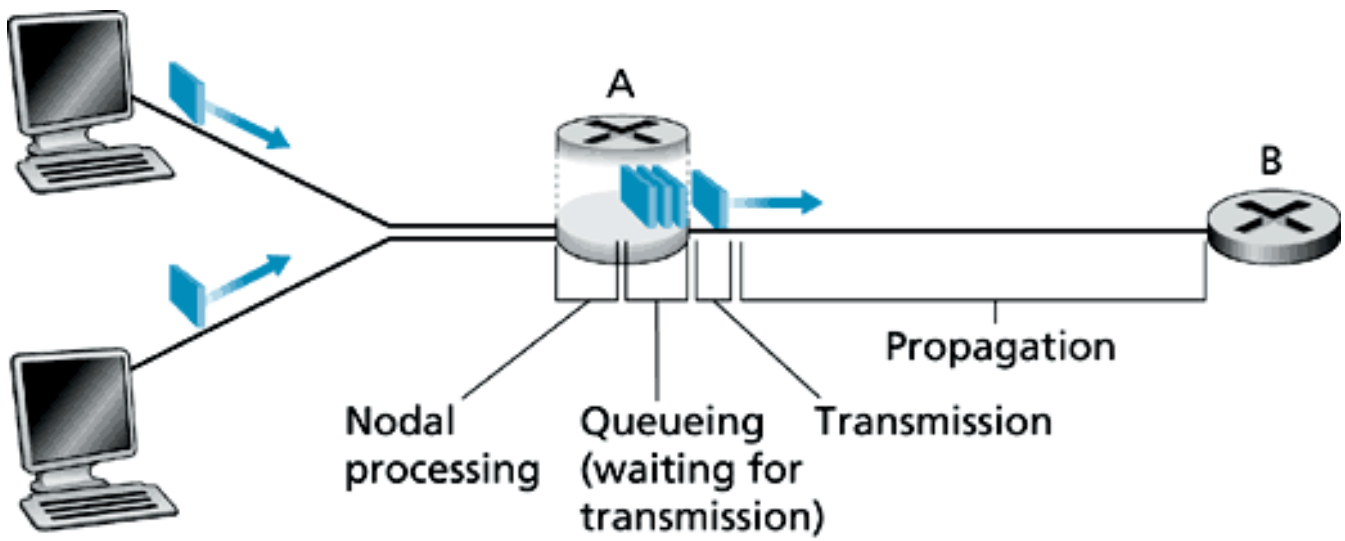
  http://www.ace.ual.es/~{}vruiz/docencia/redes/teoria/applets/queuing.html

4.4. Tiempo de transmisión (t_{tran})

- Tiempo que tardan todos los bits de un paquete (cabecera y datos) en ser introducidos en y extraídos de un enlace.
- Depende de la tasa de transmisión R y del tamaño S del paquete.

$$t_{\text{tran}} = \frac{S}{R}$$

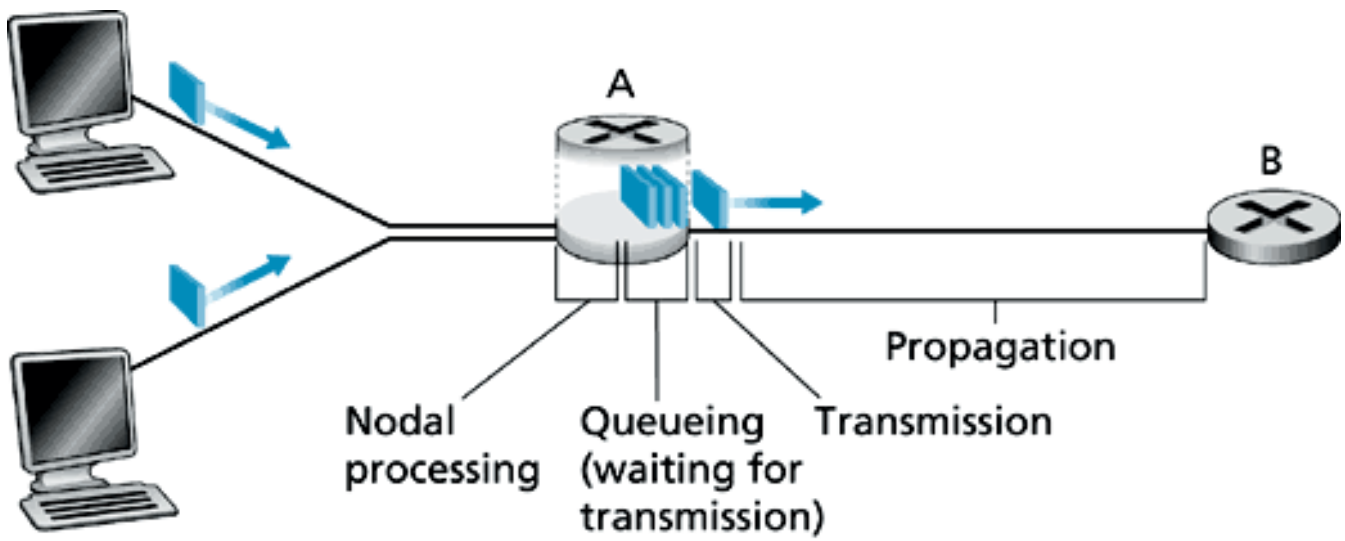
- En la práctica suele ser del orden de los milisegundos.



4.5. Tiempo de propagación (t_{prop})

- Tiempo necesario para que una señal que indica el comienzo de un bit de datos pueda propagarse desde un extremo a otro del enlace.
- Depende la distancia a recorrer D y de la velocidad C de propagación de las señales (normalmente electromagnéticas) en el medio de transmisión.

$$t_{\text{prop}} = \frac{D}{C}$$



La velocidad de propagación depende el medio.

Medio	C
Vacío/Aire	3×10^8 m/s
Cobre	$2{,}3 \times 10^8$ m/s
Fibra Óptica	2×10^8 m/s

- En WAN's puede llegar a ser del orden de milisegundos.

4.6. Tiempo de retardo nodal (t_{nodal})

- Es el tiempo que tarda un router en retransmitir un paquete hasta el siguiente router.
- Por definición es la acumulación de todos los tiempos anteriormente descritos:

$$t_{\text{nodal}} = t_{\text{proc}} + t_{\text{cola}} + t_{\text{tran}} + t_{\text{prop}}$$

- El retardo total que experimenta un paquete en cruzar la red depende del número de saltos (pasos por un router) que debe realizar. A este tiempo también se lo conoce como $t_{\text{end-to-end}}$ o retardo de extremo-a-extremo.

4.7. Pérdida de paquetes a causa de la congestión

- Los routers descartan los paquetes cuando la cola de salida de los paquetes están llenas.
- El nivel de llenado de una cola depende de la "velocidad" a la que llegan los paquetes a ella y de la "velocidad" a la que los paquetes son transmitidos. Esto en definitiva depende de las tasas de bits de llegada R_{i} y de salida R_{o}.
- Si el ratio (intensidad de tráfico)

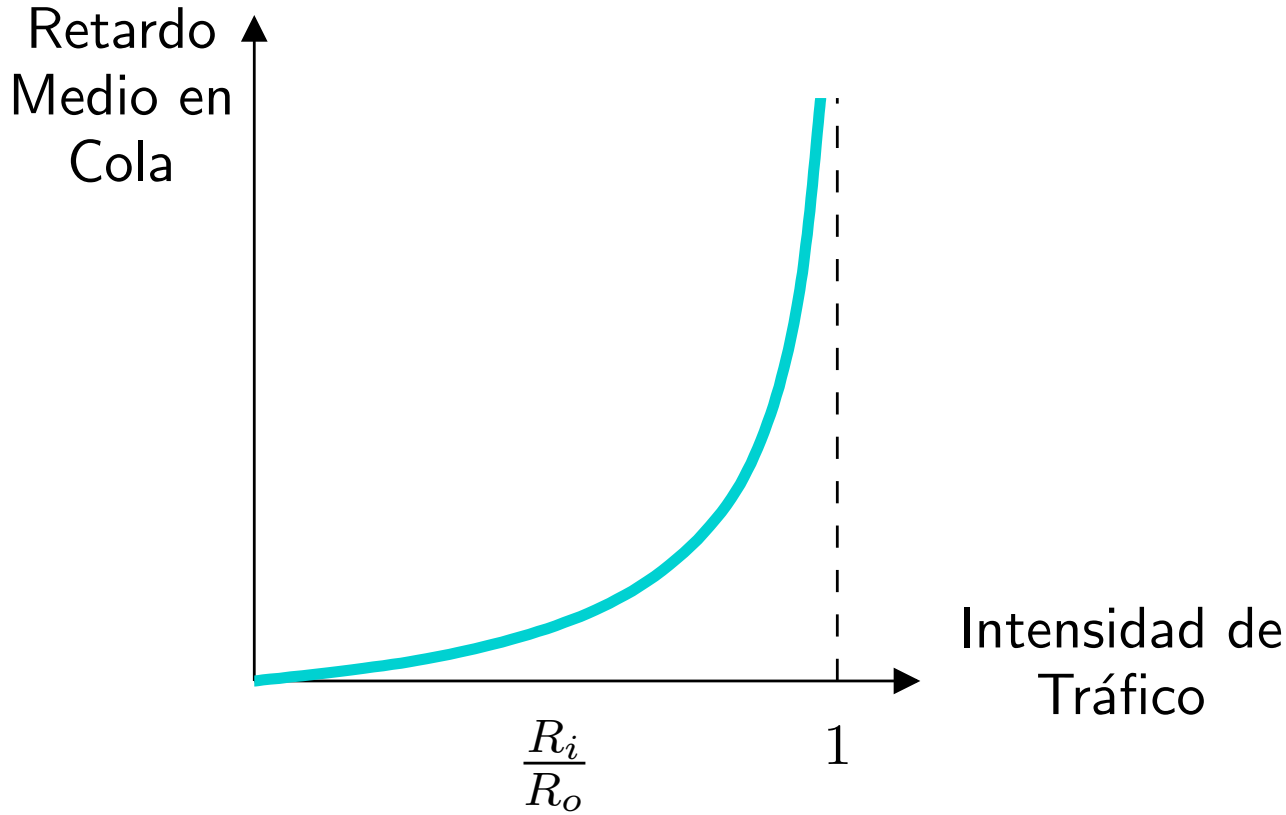
$$\frac{R_{\text{i}}}{R_{\text{o}}} > 1$$

entonces, la cola se desbordará y los paquetes comenzarán a ser desechados. Si el ratio

$$\frac{R_{\text{i}}}{R_{\text{o}}} \leq 1$$

entonces, la cola no se desbordará.

- La intensidad de tráfico y el retardo medio en las colas de salida se relacionan de la siguiente manera



lo que significa que el tiempo de transmisión de los paquetes a través de Internet depende exponencialmente de la intensidad del tráfico.

4.8. Ejemplos

1. **Enunciado:** *Sea S la longitud media de los paquetes (en bits) y α la tasa media de los paquetes que llegan hasta un router. Calcúlese la tasa de bits de llegada al router R_i.*

   Solución: S se expresa en bits/paquete y α en paquetes/segundo. Por tanto:

$$R_i = S(\frac{\text{bits}}{\text{paquete}})\alpha(\frac{\text{paquetes}}{\text{segundo}}).$$

2. **Enunciado:** *Supóngase un router con n enlaces de entrada y que al mismo tiempo llegan n paquetes (uno por cada enlace). Determínese el máximo tiempo de cola t_{cola} que sufre uno de los paquetes. Supóngase que L es la longitud media de los paquetes (en bits) y que R es la tasa de transmisión en paquetes/segundo.*

   Solución: El primer paquete servido se retrasará 0 segundos, el segundo L/R segundos, ... y el n-ésimo

$$t_{\text{cola}} = (n-1)\frac{L}{R}.$$

3. **Enunciado:** *Calcúlese el tiempo mínimo de extremo-a-extremo entre dos hosts en Internet.*

   Solución: El tiempo mínimo se determina cuando los routers están descongestionados, es decir, cuando $t_{\text{cola}} = 0$. Si N es el número de routers en el camino que une ambos extremos, entonces

$$t_{\text{end-to-end\_min}} = (N+1)(t_{\text{proc}} + t_{\text{tran}} + t_{\text{prop}}).$$

   (Téngase en cuenta el enlace que va desde el primer host origen hasta el primer router. También se ha supuesto que el host consume un t_{proc}.)

4. **Enunciado (Prob 1.6 [15]):** *Este problema elemental explora el retraso de propagación y el retraso de transmisión, dos conceptos fundamentales en las redes de datos. Considere dos hosts A y B conectados mediante un único enlace de R bps. Suponga que los dos hosts están separados por m metros, y suponga que la velocidad de propagación a lo largo del enlace es de s metros/segundo. El host A envía al host B un paquete de tamaño L bits.*

   - *Exprese el retraso de propagación, t_{prop} en términos de m y s.*

   Solución:

$$t_{\text{prop}} = \frac{m(\text{metros})}{s(\frac{\text{metros}}{\text{segundo}})} = \frac{m}{s}(\text{segundos}).$$

- *Determine el tiempo de transmisión de un paquete, t_{trans} en términos de L y R.*

Solución:

$$t_{\text{trans}} = \frac{L(\frac{\text{bits}}{\text{paquete}})}{R(\frac{\text{bits}}{\text{segundo}})} = \frac{L}{R}(\frac{\text{segundos}}{\text{paquete}}).$$

- *Ignorando los retrasos de procesamiento y de cola, obtenga una expresión para el retraso de extremo a extremo.*

Solución:

$$t_{\text{end-to-end}} = (\frac{m}{s} + \frac{L}{L})(\frac{\text{segundos}}{\text{paquete}}).$$

- *Suponga que un host A comienza a transmitir un paquete en el instante de tiempo $t = 0$. En el instante de tiempo $t = t_{\text{trans}}$, ¿dónde está el último bit del paquete?*

Solución: Acaba de abandonar el host A.

- *Suponga que* $t_{\text{prop}} > t_{\text{trans}}$. *En el instante de tiempo* $t = t_{\text{trans}}$, *¿dónde está el primer bit del paquete?*

Solución: Viajando por el enlace de transmisión.

- *Suponga que* $t_{\text{prop}} < t_{\text{trans}}$. *En el instante de tiempo* $t = t_{\text{trans}}$, *¿dónde está el primer bit del paquete?*

Solución: En el host B.

- *Suponga que* $s = 2{,}5 \times 10^8$ metros/segundo, *que* $L = 100$ bits, *y que* $R = 28$ kbps. *Encuentre la distancia* m *de manera que* $t_{\text{prop}} = t_{\text{trans}}$.

Solución: Si

$$t_{\text{trans}} = t_{\text{prop}}$$

entonces

$$\frac{m}{s} = \frac{L}{R}.$$

Sustituyendo

$$\frac{m}{2{,}5 \times 10^8 \ \frac{\text{metros}}{\text{segundo}}} = \frac{100 \text{ bits}}{28 \times 10^3 \ \frac{\text{bits}}{\text{segundo}}}.$$

Despejando

$$m = 893 \times 10^3 \text{ metros.}$$

Capítulo 5

Arquitectura del TCP/IP

5.1. El modelo de capas

- Es una forma de organización de sistemas complejos que permite implementarlos mediante subsistemas (capas) más simples [4, 8, 14].
- El modelo permite reemplazar la implementación de una de las capas sin que las demás tengan que modificarse.
- Para ello, se define un interfaz de funcionamiento de cada capa mediante el cual se intercambian datos y se solicitan servicios.

5.2. Funciones de las capas en el modelo de red

- En una red de computadoras, en general cada capa puede realizar una o más de las siguientes tareas:
  1. Control de errores de transmisión.
  2. Control de flujo de datos y de congestión.
  3. Segmentación de mensajes y reensamblado de paquetes de datos.
  4. Multiplexado de los enlaces de transmisión.
  5. Establecimiento de conexiones.

5.3. Las capas de Internet

- Internet se organiza en 5 capas:

  1. **Capa de aplicación:** en ella corren las aplicaciones de red (la Web, ftp, e-mail, etc.).
  2. **Capa de transporte:** transfiere datos entre las aplicaciones que corren en 2 o más hosts.
  3. **Capa de red:** es responsable del routing, es decir, de conducir los paquetes de datos entre los hosts.
  4. **Capa de enlace:** resuelve el problema de transmitir datos entre cada par de nodos (host-host, host-switch, host-router, router-router, etc.).
  5. **Capa física:** entiende cómo deben transmitirse los flujos de bits sobre los distintos medios físicos.

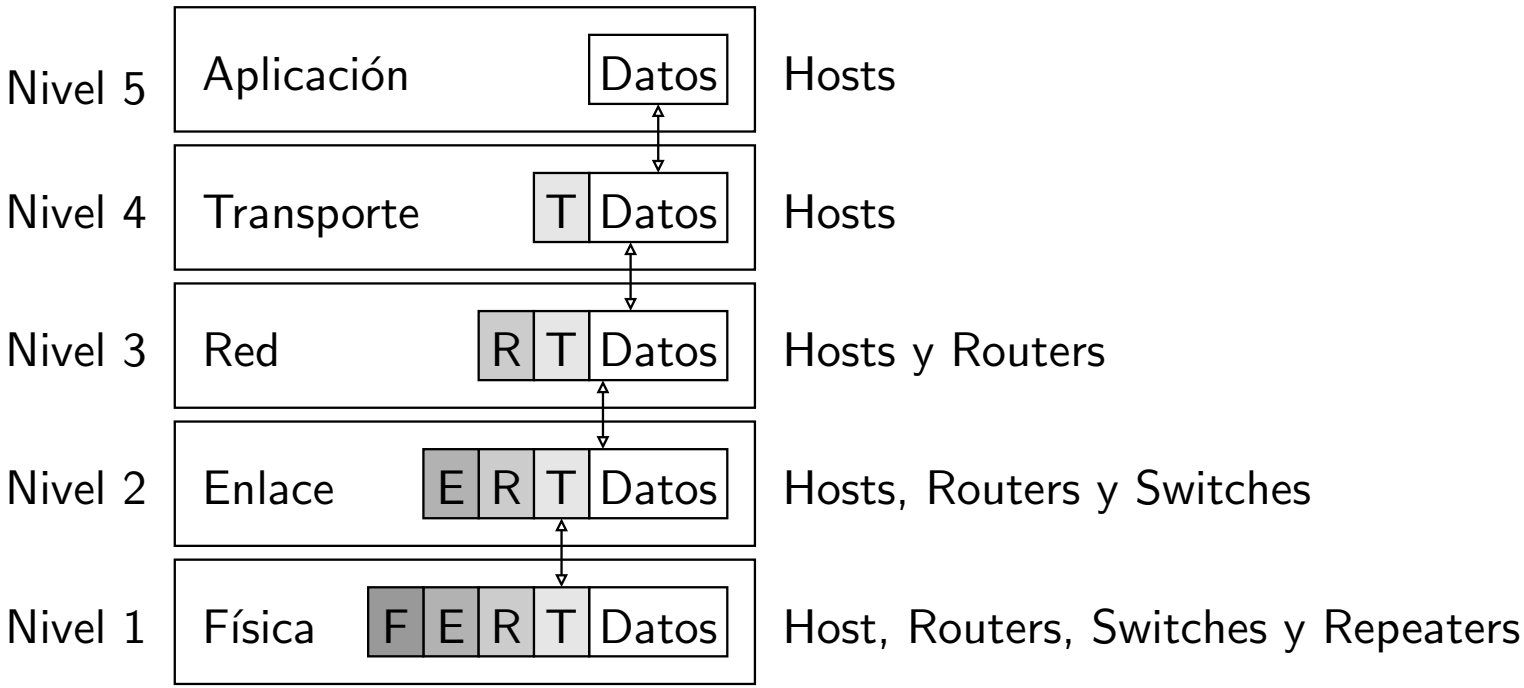
- En ocasiones y dependiendo de la complejidad, las capas pueden subdividirse en sub-capas o pueden fusionarse.

5.4. Capas y protocolos

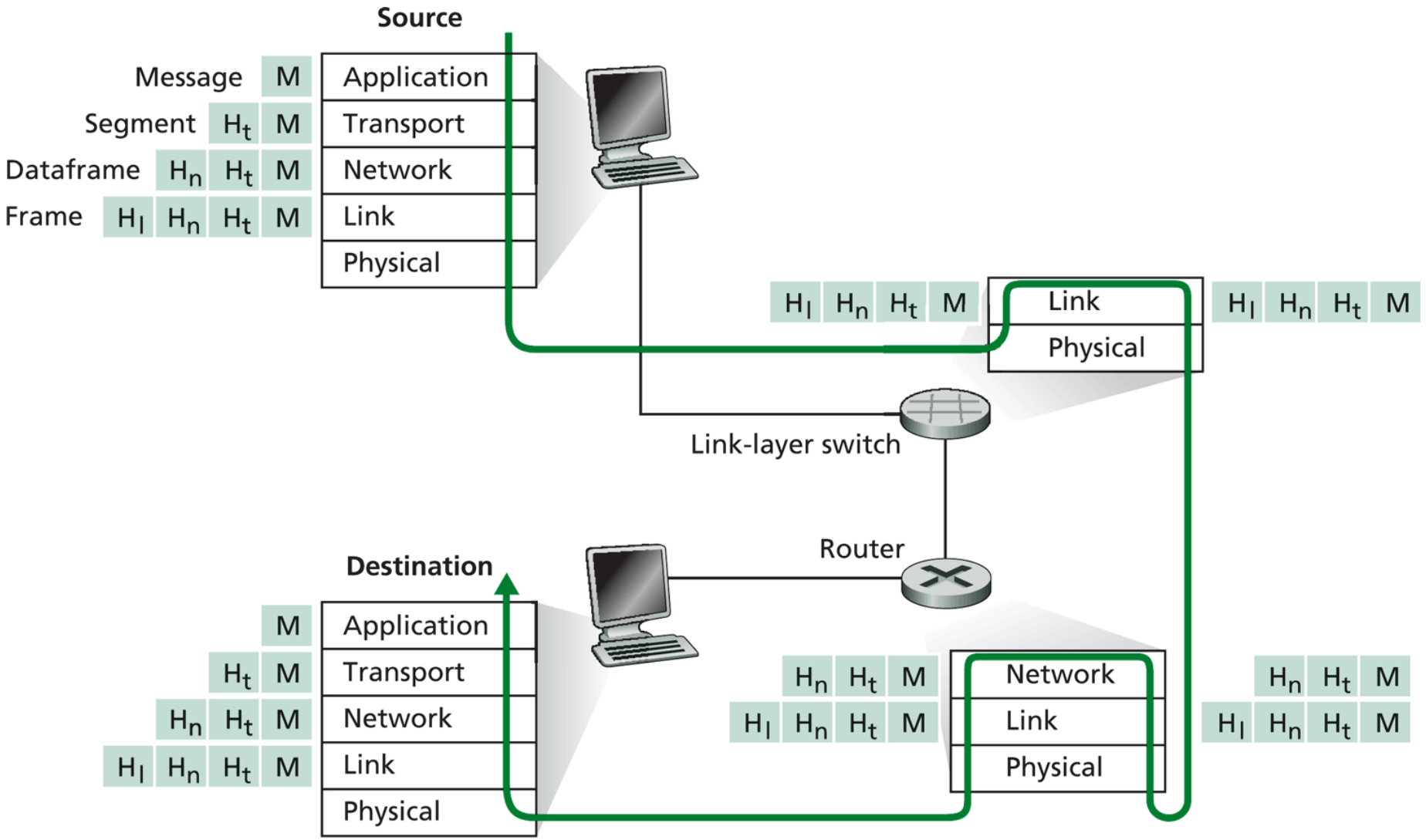
- La funcionalidad de cada capa está definida finalmente por el conjunto de protocolos que implementa.
- En Internet, los protocolos más utilizados son el TCP y el IP. Por este motivo hablamos de la pila de protocolos TCP/IP, aunque existen muchos más.
- Ordenados por capas, algunos de los protocolos más importantes son:
  1. **Capa de aplicación**: HTTP (Web), SMTP (e-mail), FTP (ftp), ...
  2. **Capa de transporte**: TCP (transferencia fiable de datos) y UDP.
  3. **Capa de red**: IP (routing), ICMP (Internet Control Message Protocol), ...
  4. **Capa de enlace de datos:** CSMA/CD (Ethernet), Point-to-Point Protocol (PPP), ...

5.5. Capas y nodos

- El número de capas que un nodo debe implementar depende de su funcionalidad.

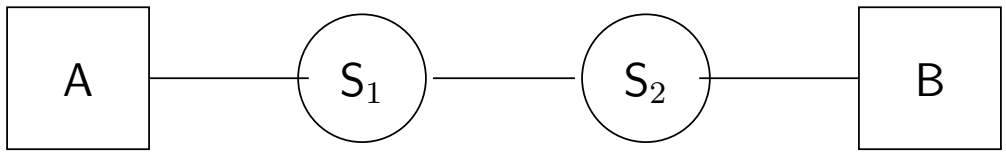


- El número de capas de enlace y física es igual al número de interfaces de red (en la figura sólo 1).
- Al proceso de añadir cabeceras y entregar el PDU (Procesing Data Unit) a la capa inferior se le llama encapsulamiento.


Source
Message M
Segment H_t M
Dataframe H_n H_t M
Frame H_l H_n H_t M
Application
Transport
Network
Link
Physical
H_l H_n H_t M
Link
Physical
H_l H_n H_t M
Link-layer switch
Router
Destination
M
H_t M
H_n H_t M
H_l H_n H_t M
Application
Transport
Network
Link
Physical
H_n H_t M
H_l H_n H_t M
Network
Link
Physical
H_n H_t M
H_l H_n H_t M

5.6. Ejemplos

1. **Enunciado (Prob 1.20 [15]):** *En las redes de conmutación de paquetes modernas el host origen segmenta los mensajes largos generados en la capa de aplicación en paquetes más pequeños antes de enviarlos a la red, y el receptor los reensambla. Nos referiremos a este proceso como la segmentación de mensajes. Supóngase una red como la de la figura:*

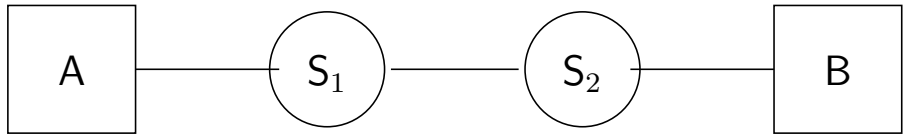


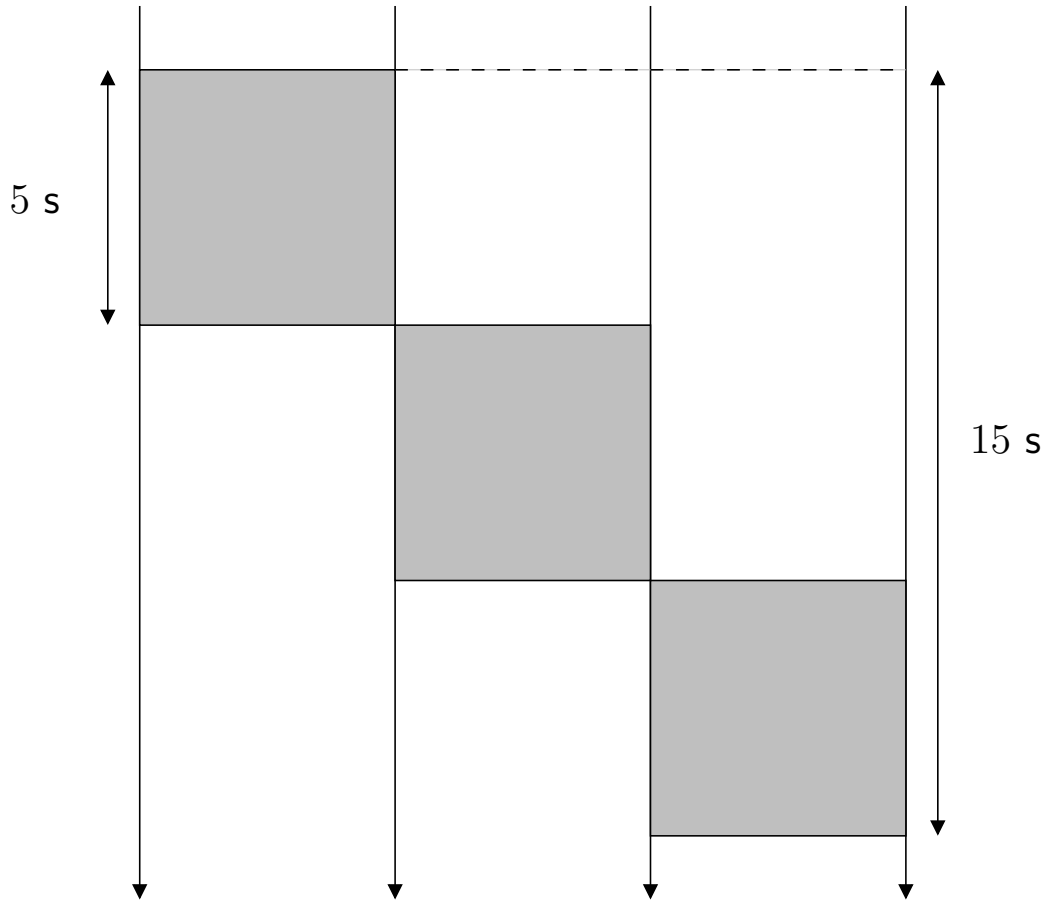
En la que se envía un mensaje de $7{,}5 \times 10^6$ *bits y la tasa de transmisión de los enlaces es de* $1{,}5 \times 10^6$ *bits/segundo.*

   - *Suponiendo que los conmutadores son del tipo store-and-forward, ¿qué tiempo transcurre desde que el host A envía el mensaje hasta que éste es recibido por el host B? Ignorense los retrasos de propagación, de cola y de procesamiento.*

Solución:

$$\frac{7{,}5 \times 10^6 \text{b}}{1{,}5 \times 10^6 \text{b/s}} = 5\text{s}.$$

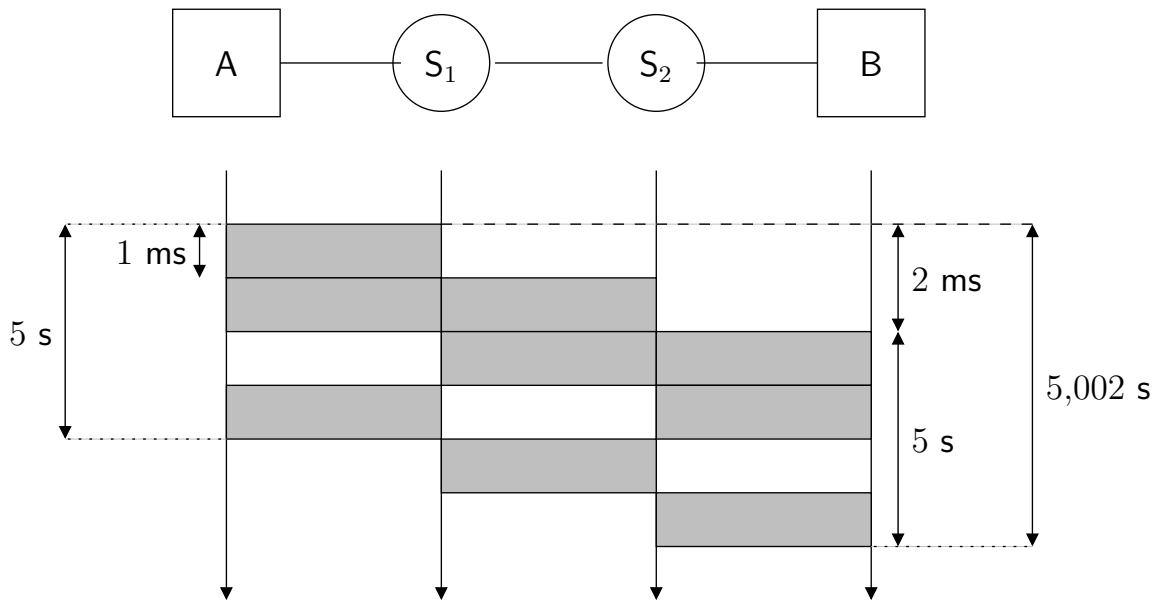




- *Suponga ahora que el mensaje se segmenta en 5000 paquetes, cada uno de 1500 bits, ¿qué tiempo transcurre desde que el host A envía el mensaje hasta que éste es recibido por el host B? De nuevo, ignórense los retrasos de propagación, de cola y de procesamiento.*

Solución:

$$\frac{1{,}5 \times 10^3 \text{b}}{1{,}5 \times 10^6 \text{b/s}} = 1\text{ms}.$$



- *Comente algunos inconvenientes de la segmentación de mensajes.*

Solución:

a) El overhead de las cabeceras.

b) La reordenación de los framentos en el receptor.

Parte II

La capa de aplicación

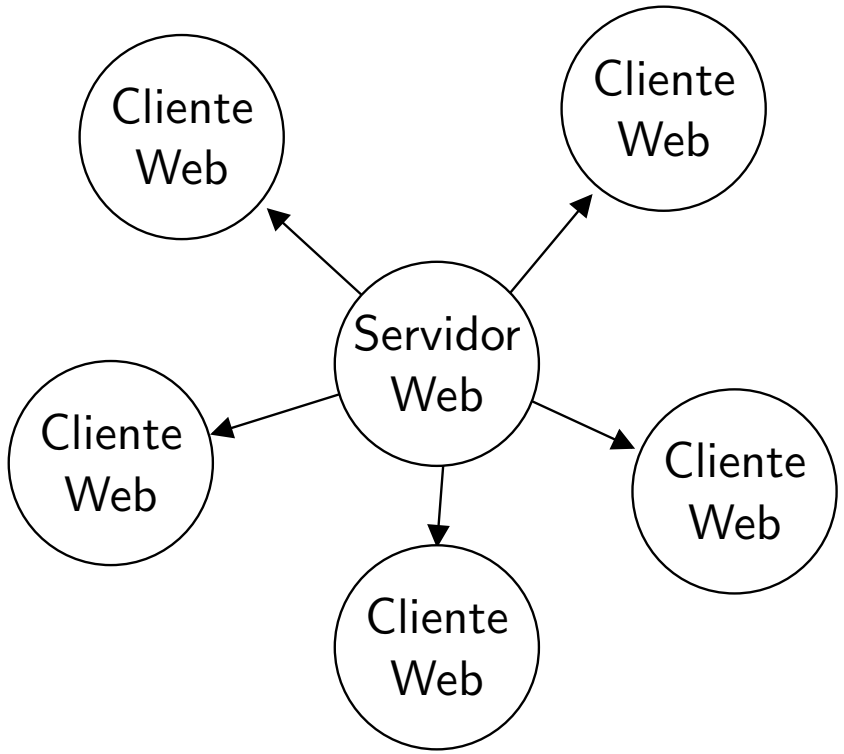
Capítulo 6

La Web

6.1. ¿Qué es la Web?

- La (World Wide) Web es una aplicación distribuida, inventada por Tim Berners-Lee [6] a principios de los 90, que permite la transmisión de información bajo demanda.

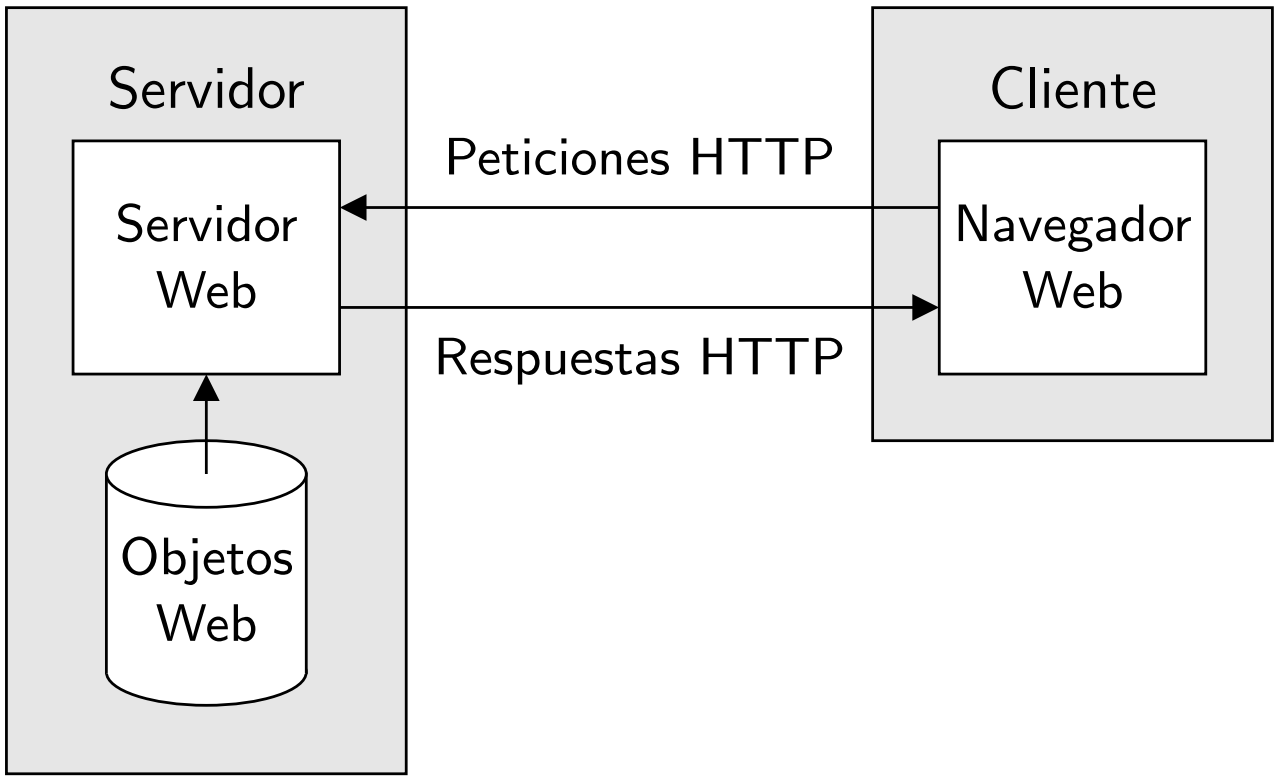
Típicamente, un servidor Web establece una comunicación de forma simultánea con muchos clientes Web. El protocolo que define el intercambio de información es el HTTP (HyperText Transfer Protocol). Este se define en los RFC's 1945 (HTTP/1.0) y 2616 (HTTP/1.1).



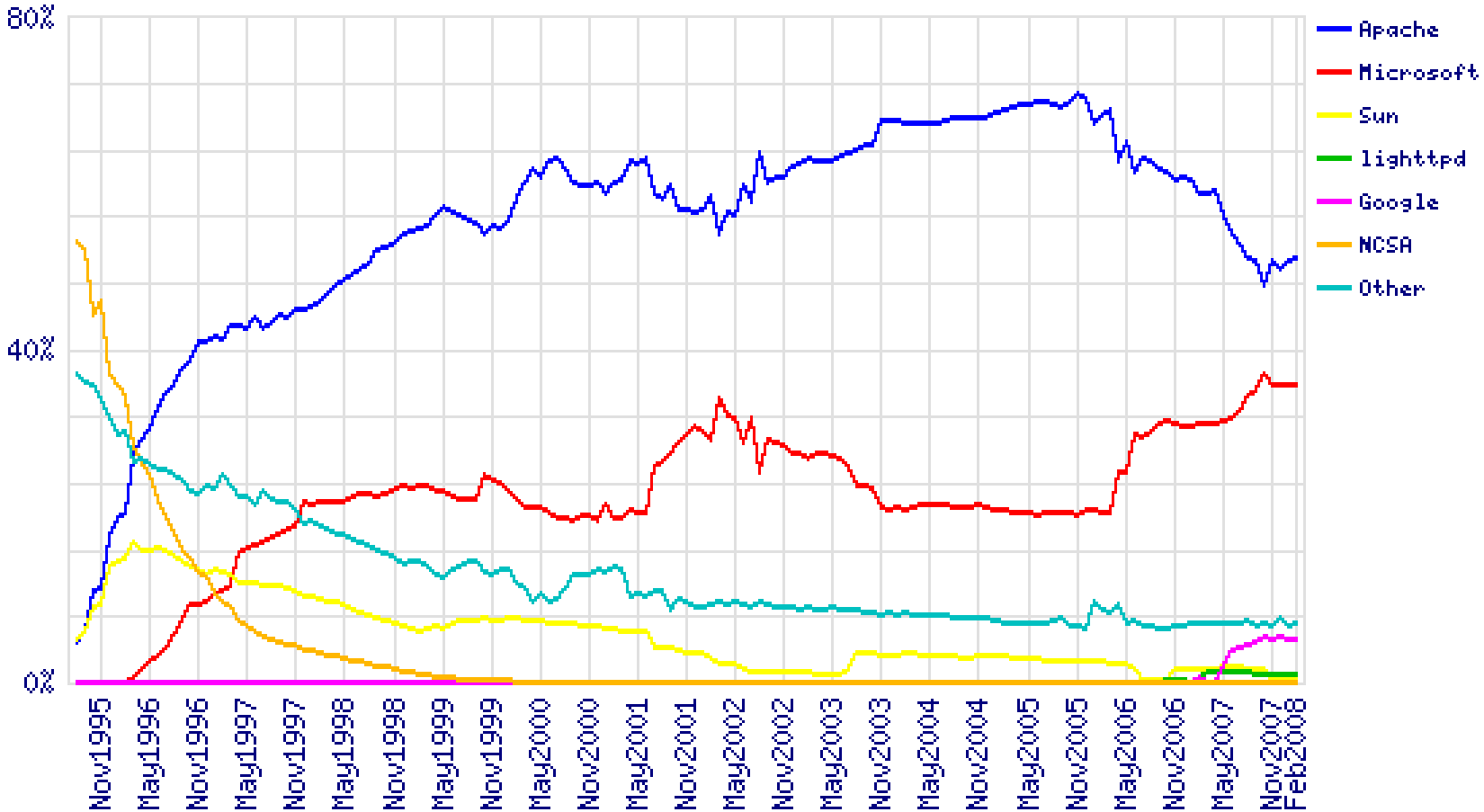
- La Web utiliza el TCP como protocolo de transporte [14].

6.2. La comunicación Web

- En una comunicación Web, un navegador Web (el cliente) realiza peticiones al servidor Web (a través del puerto 80) y éste le entrega objetos Web. Ambos tipos de mensajes se realizan según el HTTP.



- Entre los servidores Web más populares se encuentran Apache y Microsoft Internet Informacion Server (véase la URL http://news.netcraft.com/archives/web_server_survey.html).



- Un objeto Web es cualquier cosa que puede ser referenciada por su URL (Universal Resource Locator). Ejemplos típicos son las páginas Web (código HTML[1]), las imágenes GIF y las imágenes JPEG, aunque actualmente se utiliza el HTTP para transmitir cualquier tipo de archivo (con cualquier contenido).

- La estructura de una URL Web siempre es de la forma:

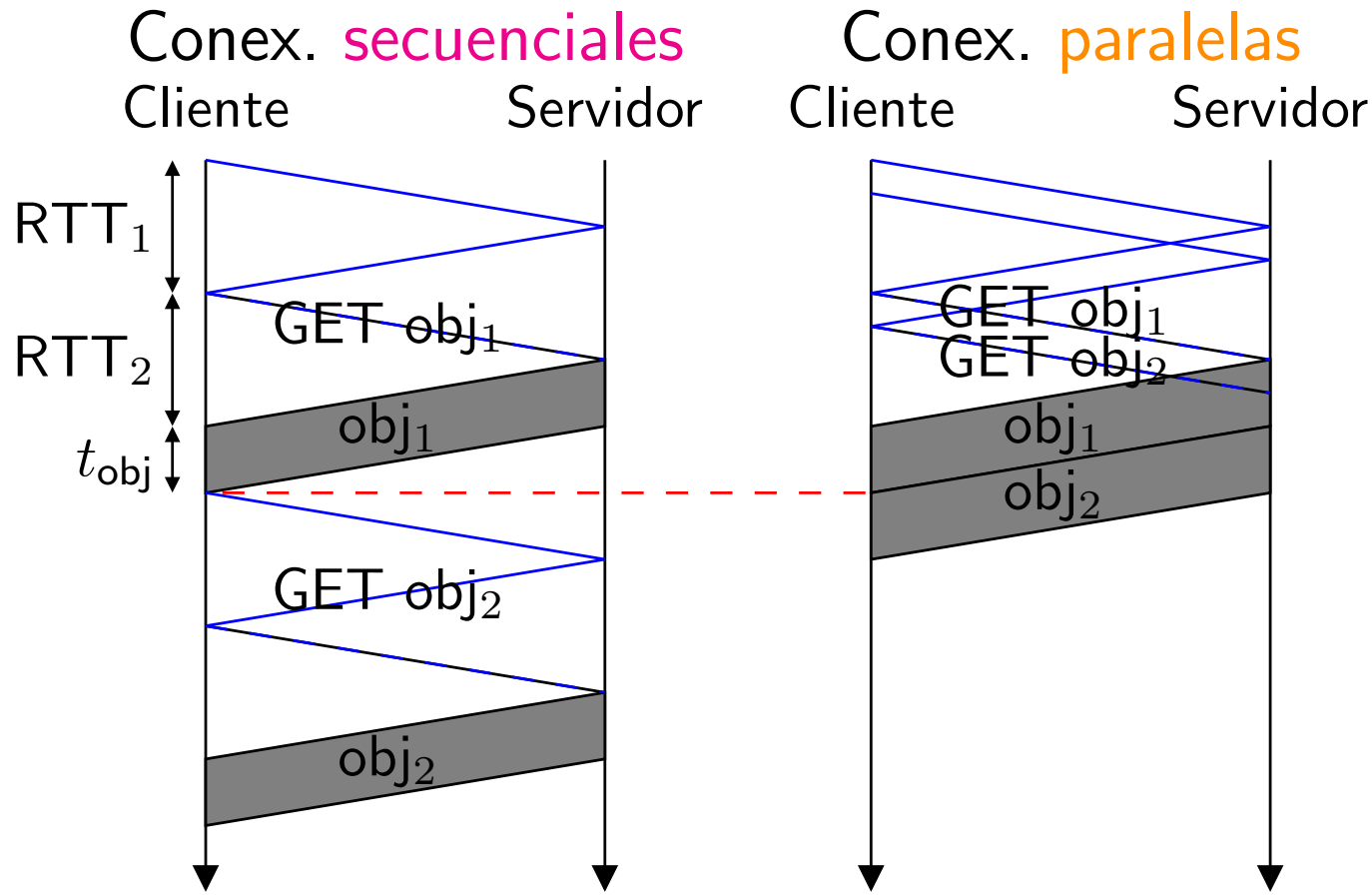
  http://host/camino_al_objeto_en_el_host

[1] HyperText Markup Language.

6.3. Las conexiones Web

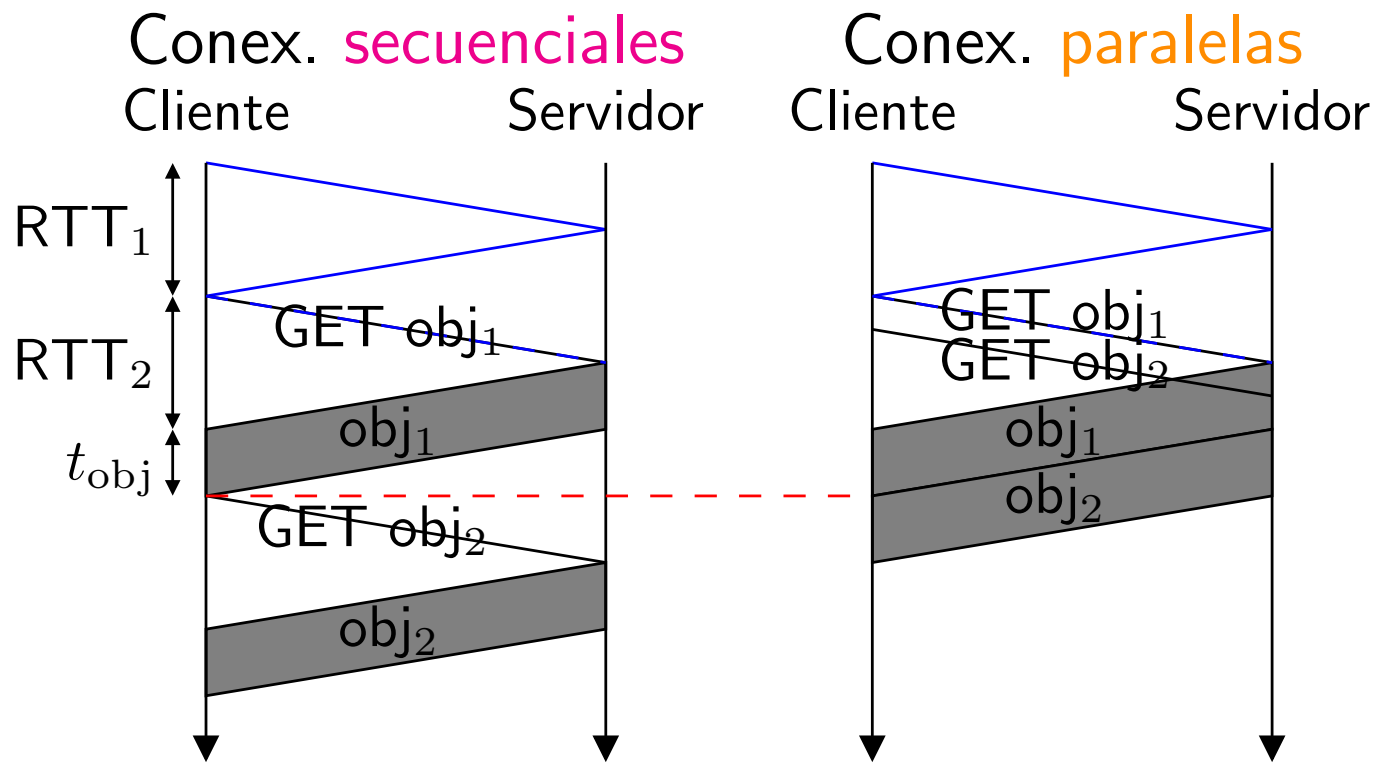
- Cuando el cliente establece una conexión HTTP con un servidor Web, ésta puede ser de 2 tipos: no persistente o persistente. El HTTP/1.0 utiliza sólo conexiones no persistentes y el HTTP/1.1 puede utilizar además conexiones persistentes.
- En una conexión no persistente, para cada objeto Web transferido se establece una conexión TCP independiente. En una conexión persistente, durante la misma conexión (establecida por un par cliente/servidor) se pueden transmitir muchos objetos Web, lo que generalmente ahorra recursos en el servidor (memoria y CPU) y en la red (ancho de banda).
- Ambos tipos de conexiones pueden ser además secuenciales o paralelas (pipelining). Estas últimas permiten ahorrar tiempo si transmitimos varios objetos Web porque el cliente puede realizar una nueva petición antes de recibir la respuesta de una previa.

6.3.1. Conexiones no persistentes



RTT (Round-Trip Time) es el tiempo de ida y vuelta de un mensaje de longitud despreciable y t_{obj} es el tiempo de transmisión del objeto Web. En cada conexión TCP, el tiempo de establecimiento de conexión TCP necesita un RTT (RTT_1). El segundo RTT (RTT_2) se emplea en solicitar el objeto.

6.3.2. Conexiones persistentes



El ahorro del establecimiento de una conexión TCP para cada conexión Web reduce el tiempo en el caso de las conexiones secuenciales. El tiempo de respuesta de las conexiones paralelas persistentes es el mismo que el de las conexiones no persistentes, aunque la carga para el servidor suele ser menor.

6.4. Los mensajes HTTP

En una comunicación HTTP sólo existen dos tipos de mensajes, los de petición (request) y los de respuesta (reply).

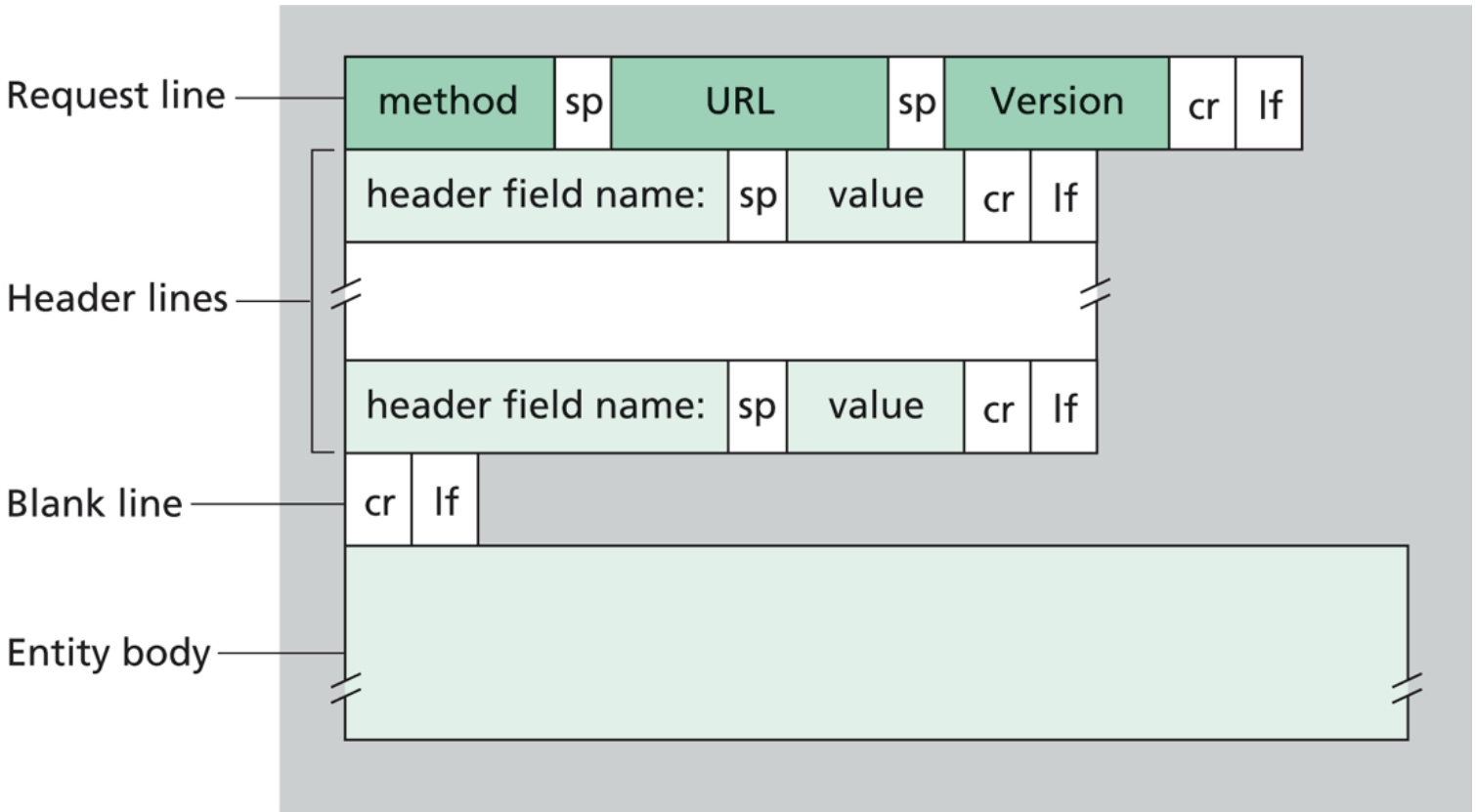


Figure 2.8 ♦ General format of a request message

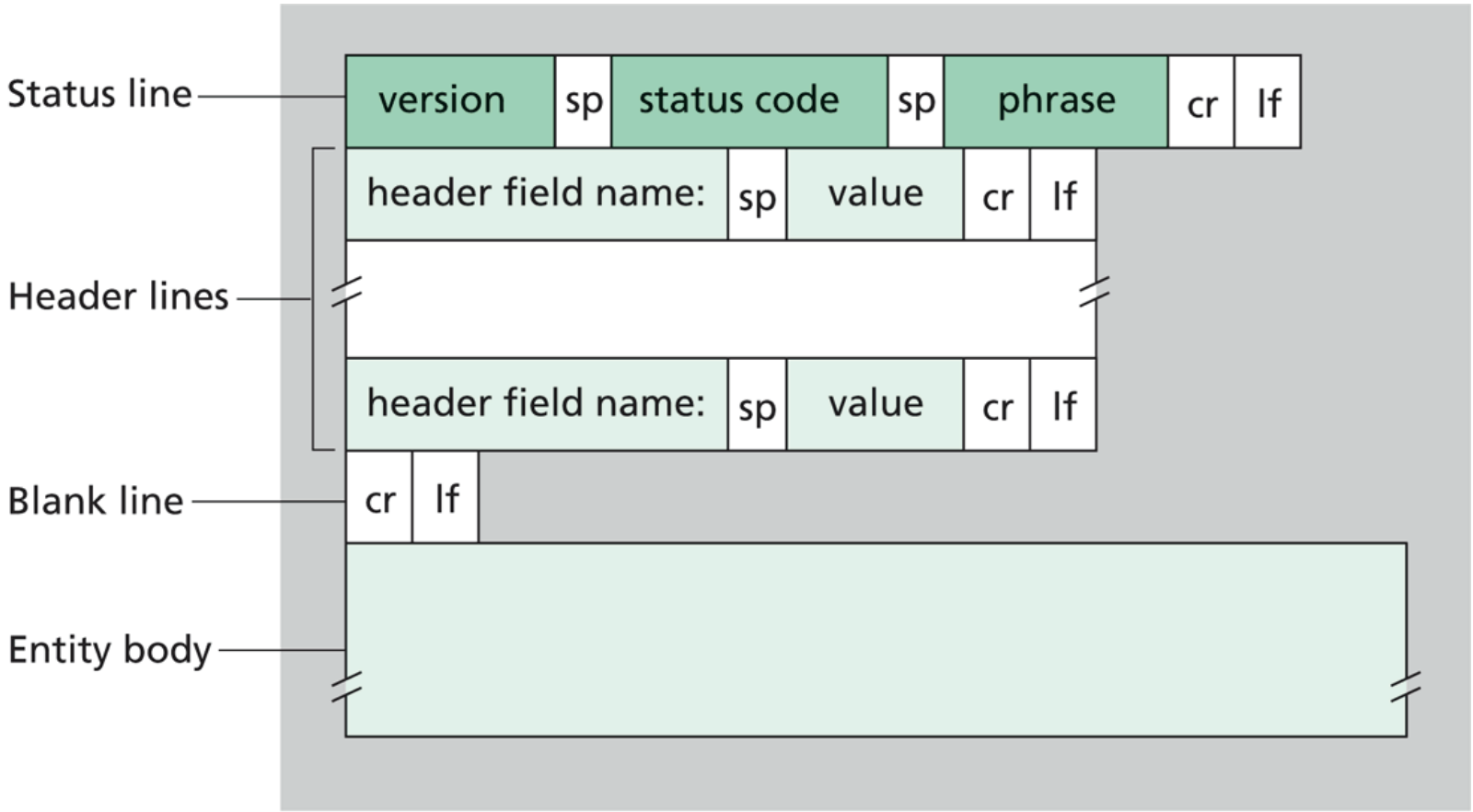


Figure 2.9 ♦ General format of a response message

6.4.1. Un mensaje de petición

Capturado usando ethereal conectándose a www.google.es mediante mozilla.

```
GET / HTTP/1.1
Host: www.google.es
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.2.1) G
Accept: text/xml,application/xml,application/xhtml+xml,text/htm
Accept-Language: en-us, en;q=0.50
Accept-Encoding: gzip, deflate, compress;q=0.9
Accept-Charset: ISO-8859-1, utf-8;q=0.66, *;q=0.66
Keep-Alive: 300
Connection: keep-alive
Cookie: PREF=ID=1e9556644260c793:LD=es:TM=1076751818:LM=107675

<Cuerpo de entidad>
```

Significado de algunas de las líneas:

- Los mensajes de petición están codificados en ASCII, comienzan siempre por la cadena `GET`, la cadena `POST` o la cadena `HEAD` y acaban siempre en una línea en blanco (retorno de carro y nueva línea).
- El número de líneas es variable, pero como mínimo siempre existe la primera (línea de petición) donde se indica el tipo de petición (`GET`), la páqina HTML reclamada (`/directorio/pagina.html`, aunque en el ejemplo se trata de la página `index.html` que es la página por defecto) y la versión del protocolo HTTP utilizado (`HTTP/1.1`).
- El resto de las líneas de cabecera.
  - La primera línea, que comienza en el ejemplo por `Host:` especifica el host en que reside el objeto (necesario para los proxies).
  - La línea `Connection:` indica el tipo de conexión (`keep-alive` significa conexión persistente y `close` conexión no persistente.
  - La línea `User-Agent:` indica el navegador Web utilizado. Esto es importante para el servidor porque dependiendo del navegador

se pueden enviar páginas HTML diferentes (con idéntica URL).

- La línea `Accept-Language:` indica que el usuario prefiere recibir una versión en inglés del objeto.

- `<Cuerpo de entidad>` es un campo de los mensajes de petición que está vacío cuando se utiliza el método `GET`, pero que sí se utiliza con el método `POST`. Este método se emplea, por ejemplo, para rellenar formularios (donde el cliente debe enviar información al servidor). Finalmente, el método `HEAD` es similar al método GET, pero el servidor en su respuesta no va a incluir el objeto pedido. Este método es normalmente utilizado por los desarrolladores de aplicaciones.

- En la versión HTTP/1.1, además de `GET`, `POST` y `HEAD`, existen otros dos métodos: `PUT` que permite a un usuario cargar un objeto en el servidor Web y `DELETE` que permite borrarlo.

6.4.2. Un mensaje de respuesta

```
HTTP/1.1 200 OK
Cache-control: private
Content-Type: text/html
Content-Encoding: gzip
Server: GWS/2.1
Content-length: 1484
Date: Sat, 14 Feb 2004 10:52:44 GMT

<Cuerpo de entidad>
```

Significado de algunas de las líneas:

- Los mensajes de respuesta siempre tienen 3 secciones: la línea inicial de estados, las lineas de cabecera y el cuerpo de la entidad.
- La línea inicial de estados tiene 3 campos: la versión del protocolo, el código de estado y el estado (estas dos cosas significan lo mismo). En el ejemplo, `200 OK` significa que el servidor ha encontrado el objeto y que lo ha servido (en el ejemplo no se muestra tal cual, se trata de

un objeto comprimido). En la siguiente tabla se presentan algunos de los códigos de control más comunes:

Código	Significado
`200 OK`	Petición exitosa
`301 Moved Permanently`	El objeto demandado ha sido movido a la URL especificada en `Location:`
`400 Bad Request`	Petición no entendida por el servidor
`404 Not Found`	Objeto no encontrado en el servidor
`505 HTTP Version Not Supported`	Obvio

- `Server:` indica el software del servidor Web.
- `Date:` indica la fecha y hora en la que se creó y envió la respuesta HTTP. Este campo es importante porque ayuda a los proxies a mantener sus cachés.
- `Content-length:` indica el tamaño en bytes del objeto enviado.

- `Content-Type:` indica que el objeto enviado en el cuerpo de la entidad es texto HTML.

6.5. Paso de parámetros en las URL's

- En la sección anterior hemos visto que uno de los métodos usados en los mensajes de petición (POST) era utilizado para enviar datos de tipo "formulario" al servidor. Esto también ser puede realizar mediante GET.
- Para ello, en la URL usada se especifican dichos datos mediante la forma:

  URL?primer parámetro&segundo parámetro&...

  Por ejemplo:

  `http://www.google.es/language_tools?hl=es`

6.6. Identificación de usuarios

- El HTTP no tiene estado lo que significa que un servidor no guarda información acerca de los datos enviados a un determinado cliente. Esto simplifica el diseño de los servidores.
- A menudo, el servidor necesita identificar a los usuarios, bien porque el acceso al servidor esté restringido, o porque quisiera servir cierto contenido en función de la identidad del usuario.
- El HTTP proporciona dos mecanismos:
  1. La autorización "login/password".
  2. Las cookies (galletas).

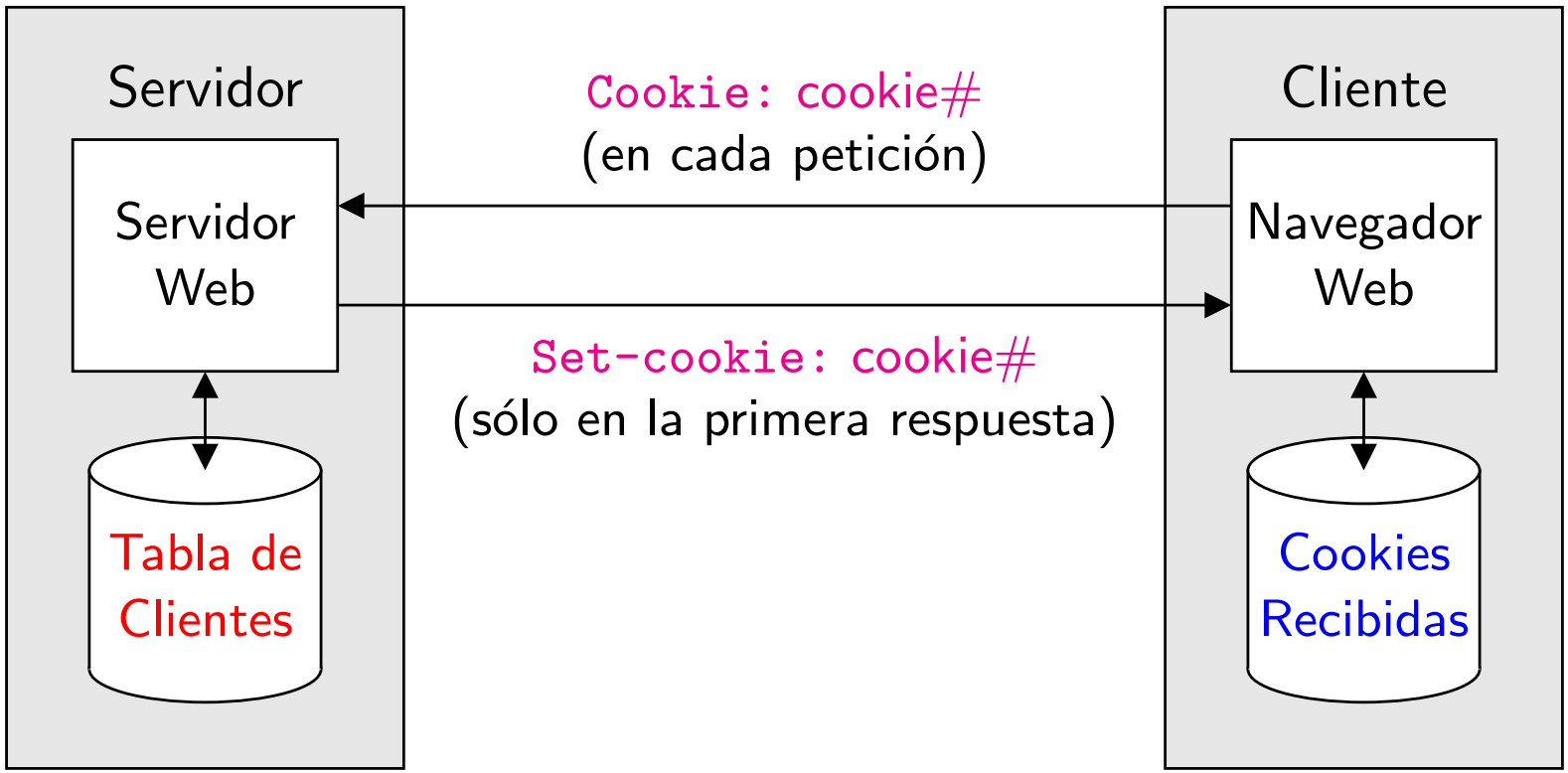
6.6.1. Autorización "login/password"

- El usuario se identifica mediante un nombre de usuario (login) y una palabra clave (password). Pasos:

  1. El servidor avisa al navegador que debe identificarse mediante una línea inicial de estado `HTTP/1.1 401 AuthorizationRequired` en su mensaje HTTP de respuesta.
  2. El navegador solicita al usuario el login/password e incluye esta información en una línea `Authorization:` del próximo mensaje de petición. Dicha línea se incluye en cualquier nueva petición de cualquier nuevo objeto al servidor.

6.6.2. Cookies

- Las cookies sirven para que los navegadores identifiquen a los usuarios que anteriormente se han conectado.



- En la tabla de clientes existen entradas de la forma (cookie#,usuario), que son creadas la primera vez que el usuario se conecta.

- En el fichero de cookies recibidas el navegador almacena una entrada (servidor Web, cookie#) cada vez que se conecta (por primera vez) a un servidor que utiliza cookies.

6.7. El GET condicional

- Los navegadores Web poseen una caché donde almacenan los objetos más recientes.
- Cuando un navegador va a reclamar un objeto, primero mira si está en su caché. Si está, entonces su petición es condicional. Si no está, su petición no es condicional.
- Cuando se realiza una petición condicional, el servidor Web envía una nueva versión sólo si la copia local es obsoleta.

6.7.1. GET (normal)

Petición

```
GET /fruit/kiwi.gif HTTP/1.0
User-agent: Mozilla/4.0
```

Respuesta

```
HTTP/1.0 200 OK
Date: Web, 12 Aug 1998 15:39:29
Server: Apache/1.3.0 (Unix)
Last-Modified: Mon, 22 Jun 1998 09:23:24
Content-Type: image/gif

(cuerpo de entidad)
```

6.7.2. GET condicional

En la caché del cliente existe el objeto reclamado.

Petición

```
GET /fruit/kiwi.gif HTTP/1.0
User-agent: Mozilla/4.0
If-modified-since: Mon, 22 Jun 1998 09:23:24
```

Respuesta

Suponiendo que el objeto no ha sido modificado desde `22 Jun 1998 09:23:24`.

```
HTTP/1.0 304 Not Modified
Date: Web, 19 Aug 1998 15:39:29
Server: Apache/1.3.0 (Unix)
```

6.8. Las cachés Web (proxies Web)

Funcionamiento

- La Web es un entramado de servidores de contenidos y de clientes (navegadores Web y cualquier otra aplicación distribuida que utilice el HTTP para comunicarse) (véase http://www.rediris.es/si/cache).
- Para minimizar los tiempos de respuesta y el tráfico en la red, la Web utiliza un conjunto de servidores especiales (llamdos proxies Web[2]) que funcionan como una caché, almacenando todo lo que sirven a los clientes para su posterior reenvío según una determinada política (por ejemplo, almacenando siempre los objetos más frecuentes).
- De esta forma, cuando un cliente (configurado para utilizar un proxy) solicita un objeto a un servidor Web, en lugar de hacerlo directamente al servidor lo realiza al proxy. El proxy entonces mira si posee el objeto

[2] Existen otros tipos de proxies que no son Web y por lo tanto, cuando exista confusión debería especificarse. En este documento esto no es así por lo que prescindiremos de especificar el tipo de proxy.

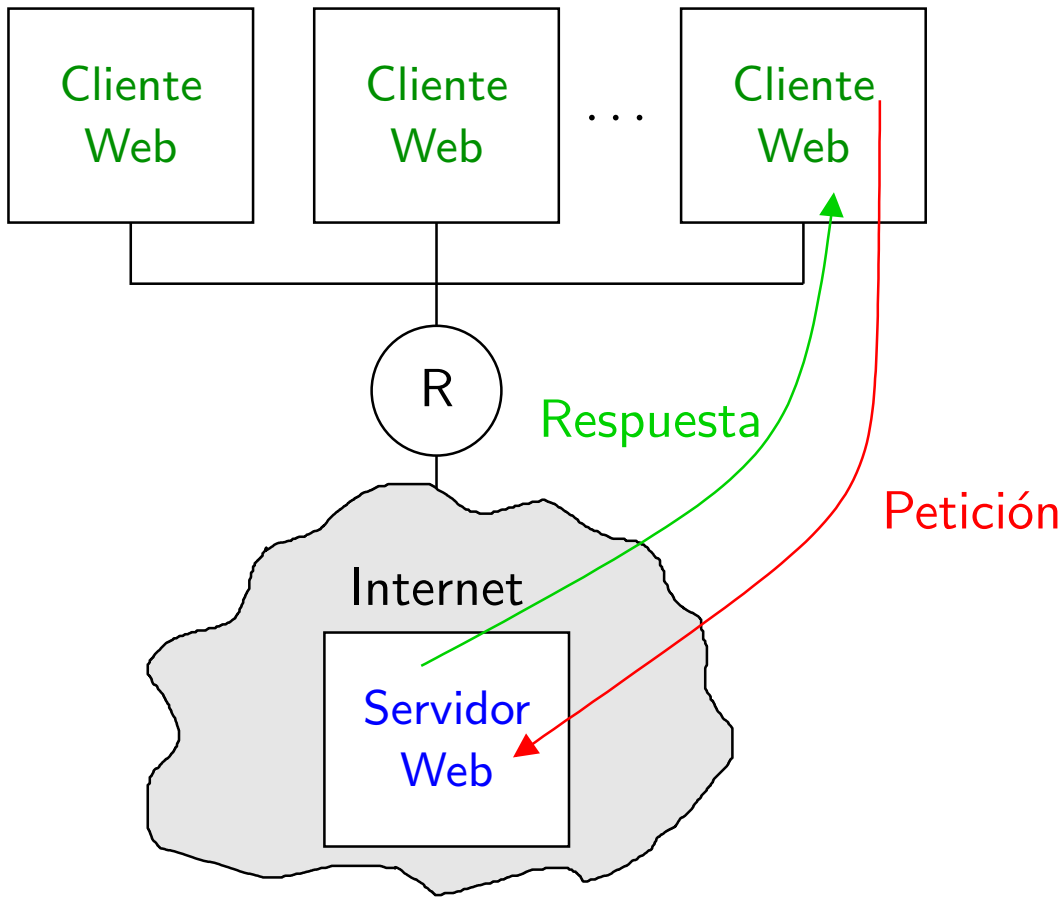
y si es así, lo sirve. En caso contrario lo solicita al servidor, lo almacena y lo sirve.

- Para mantener actualizada la caché de un proxy, este utiliza el GET condicional con el servidor (o el proxy de nivel superior).

6.9. Arquitecturas Web

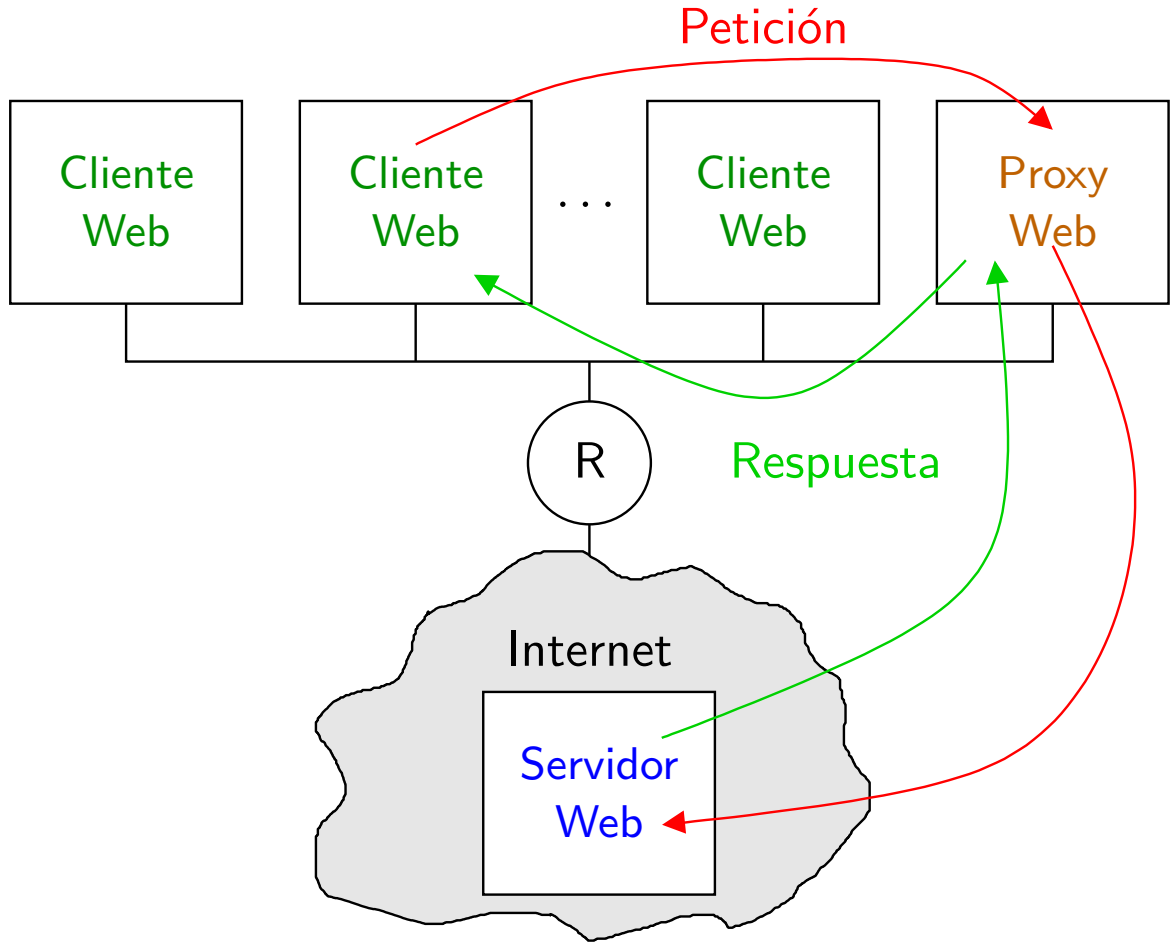
6.9.1. La configuración más sencilla

- En su configuración más simple, los clientes "atacan" directamente a/los servidores Web.



6.9.2. Sistemas proxy de 1 nivel

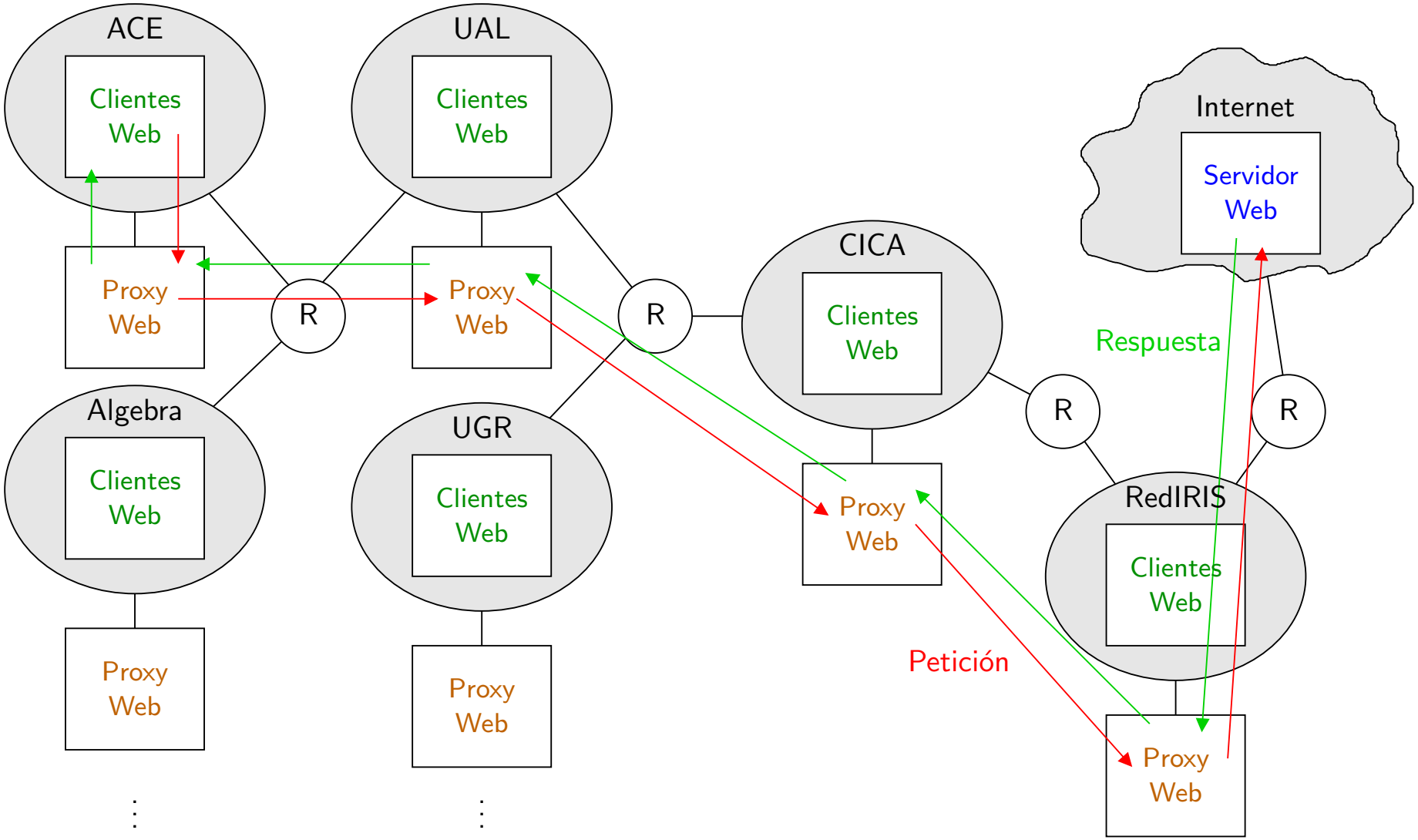
- El cliente solicita un objeto a su proxy local y si este no lo encuentra, lo solicita al servidor Web pertinente.



6.9.3. Sistemas proxy multinivel

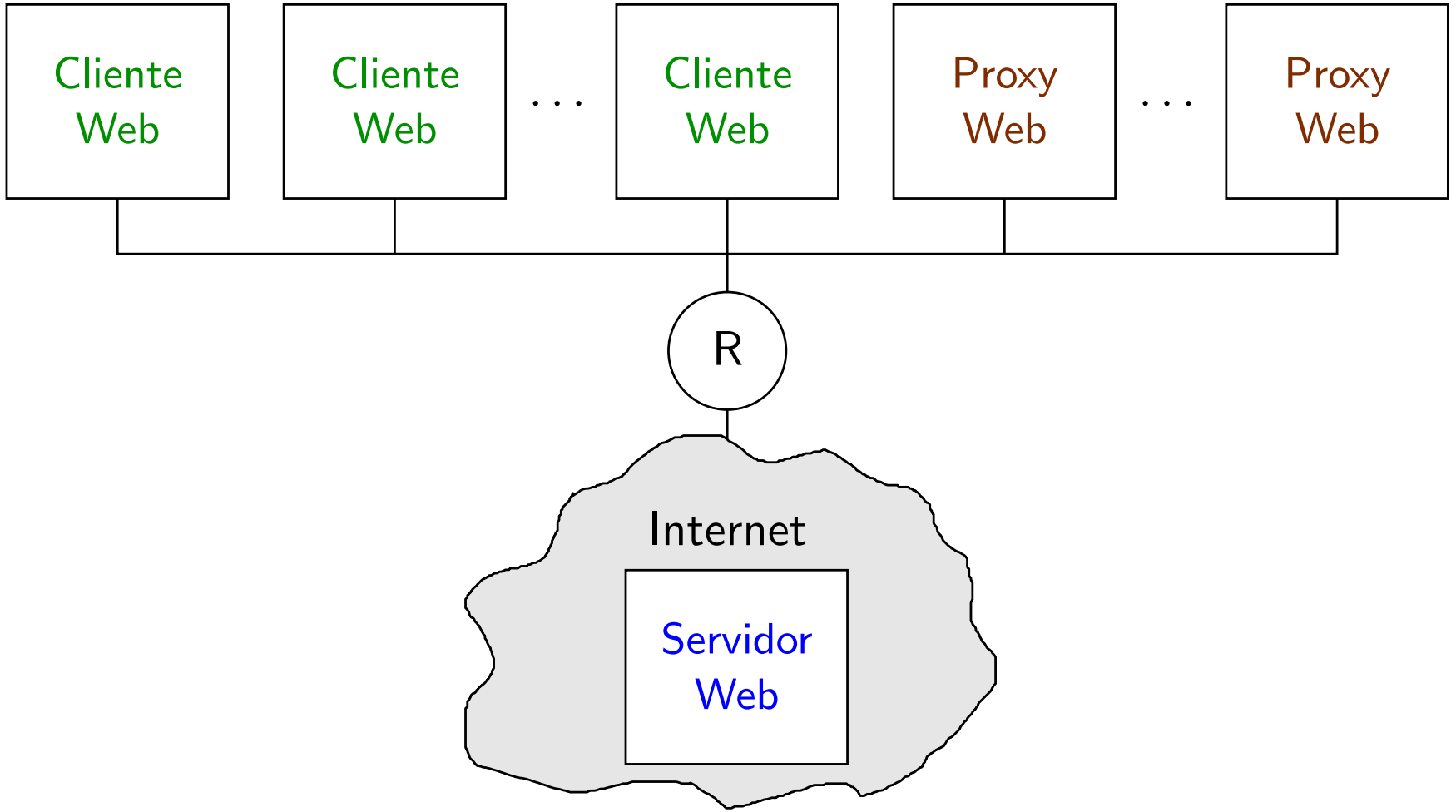
- Los servidores proxy pueden configurarse para utilizar de forma recursiva otros servidores proxy de mayor ambito. A esto se le llama caché Web cooperativa.

- Los proxies de una jerarquía pueden utilizar el Internet Caching Protocol (ICP) para hablar todos con todos a la hora de localizar un objeto dentro de la jerarquía. De esta manera, si el objeto se localiza en un proxy cercano el proceso de descarga del objeto (que finalmente se realiza mediante HTTP) se aceleraría.


ACE
Clientes Web
UAL
Clientes Web
Internet
Servidor Web
Proxy Web
R
Proxy Web
R
CICA
Clientes Web
Respuesta
R
R
Algebra
Clientes Web
UGR
Clientes Web
Proxy Web
RedIRIS
Clientes Web
Petición
Proxy Web
Proxy Web
Proxy Web
Resto de Departamentos de la UAL
Resto de Universidades Andaluzas

6.9.4. Sistemas proxy distribuidos

- Los proxies pueden soportar cargas muy altas cuando "representan" a una gran cantidad de servidores Web.
- Cuando un proxy soporta demasiada carga se puede proceder a la distribución del contenido de su caché sobre un conjuto de proxies.
- El contenido de cada proxy se controla mediante el Cache Array Routing Protocol (CARP) que utiliza rutado por dispersión (hashing). Esta forma de rutado consiste en usar un proxy distinto en función de la URL del objeto Web solicitado.
- Cuando todos los clientes utilizan la misma función de dispersión, un objeto sólo reside en uno de los proxies (en ese nivel).


Cliente Web
Cliente Web
Cliente Web
Proxy Web
Proxy Web
R
Internet
Servidor Web

Capítulo 7

El correo electrónico

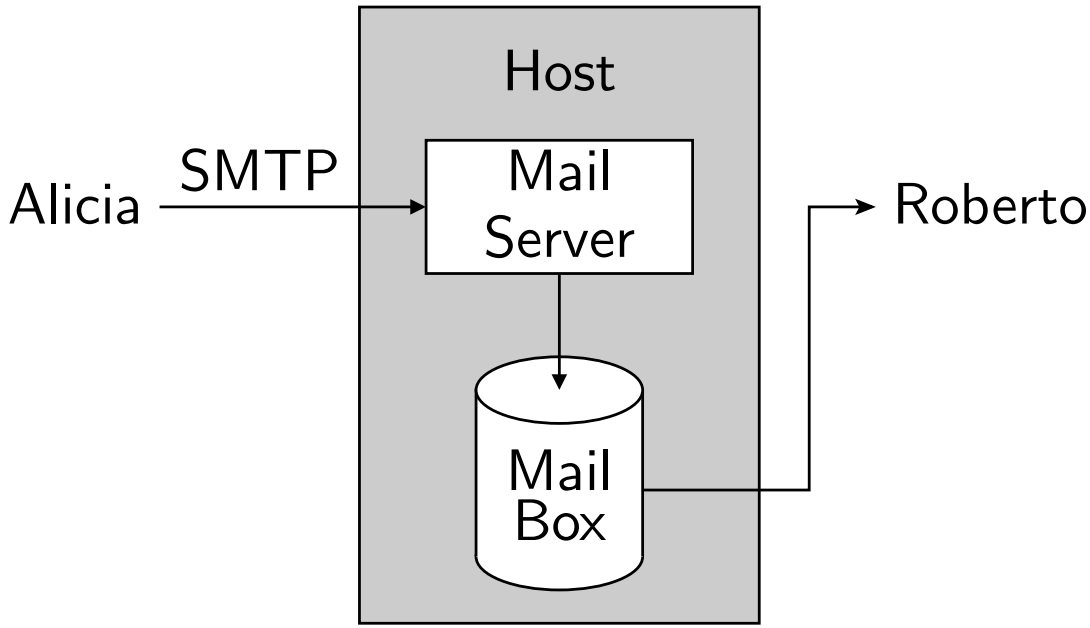
7.1. El correo electrónico

- Es un **medio de comunicación asíncrono**, mediante el cual usando Internet podemos enviar correos electrónicos a otras personas sin necesidad de una advertencia previa [14].
- En sus versiones más iniciales sólo permitía transmitir **mensajes** en formato **ASCII**. Ahora también puede transmitir cualquier otro tipo de información (**datos binarios**, **programas**, **vídeos**, **etc**.).
- Se trata de una aplicación cliente/servidor. El "cliente" envía los mensajes (e-mails) y el "servidor" los recibe.
- La comunicación cliente-servidor se realiza mediante el **SMTP** (Simple Mail Transfer Protocol).

7.2. Configuraciones

Supongamos que Alicia le envía un e-mail a Roberto. Podrían darse (entre otras) estas configuraciones:

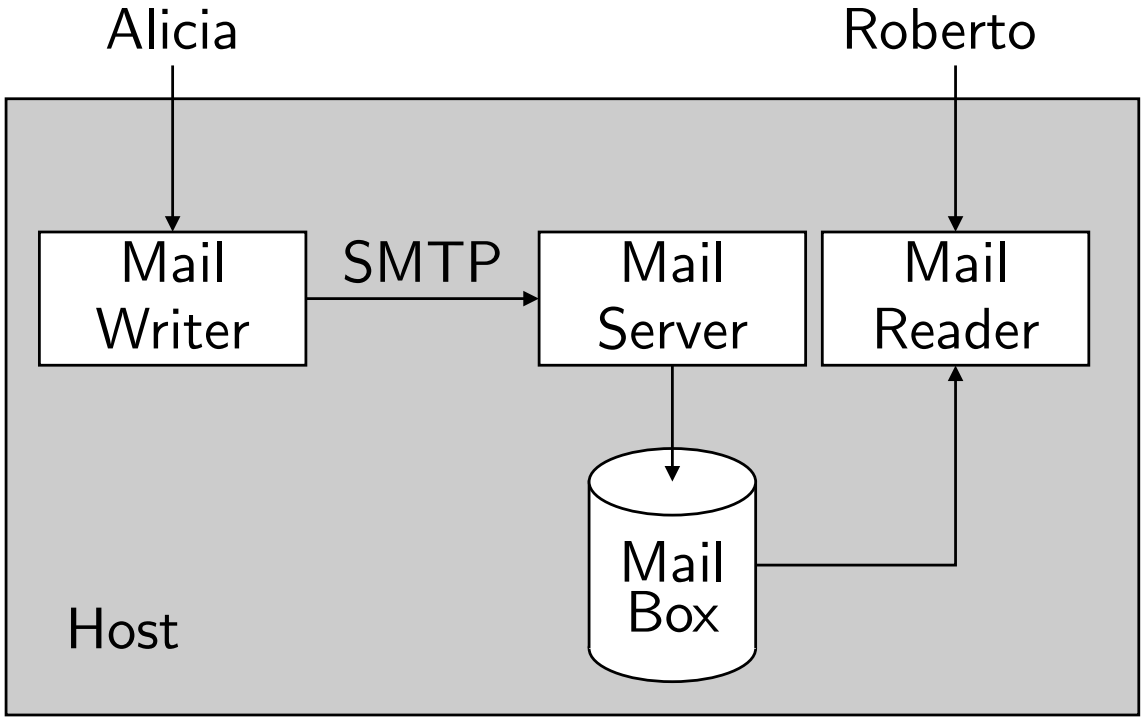
7.2.1. Correo local usando SMTP



- Alicia sabe SMTP y roberto lee el correo directamente desde su fichero de mailbox[1], donde se encolan los e-mails que se reciben y todavía no se han leído.
- En este ejemplo, Roberto utiliza un editor de ficheros ASCII.
- Ambos se comunican a través del mismo host.

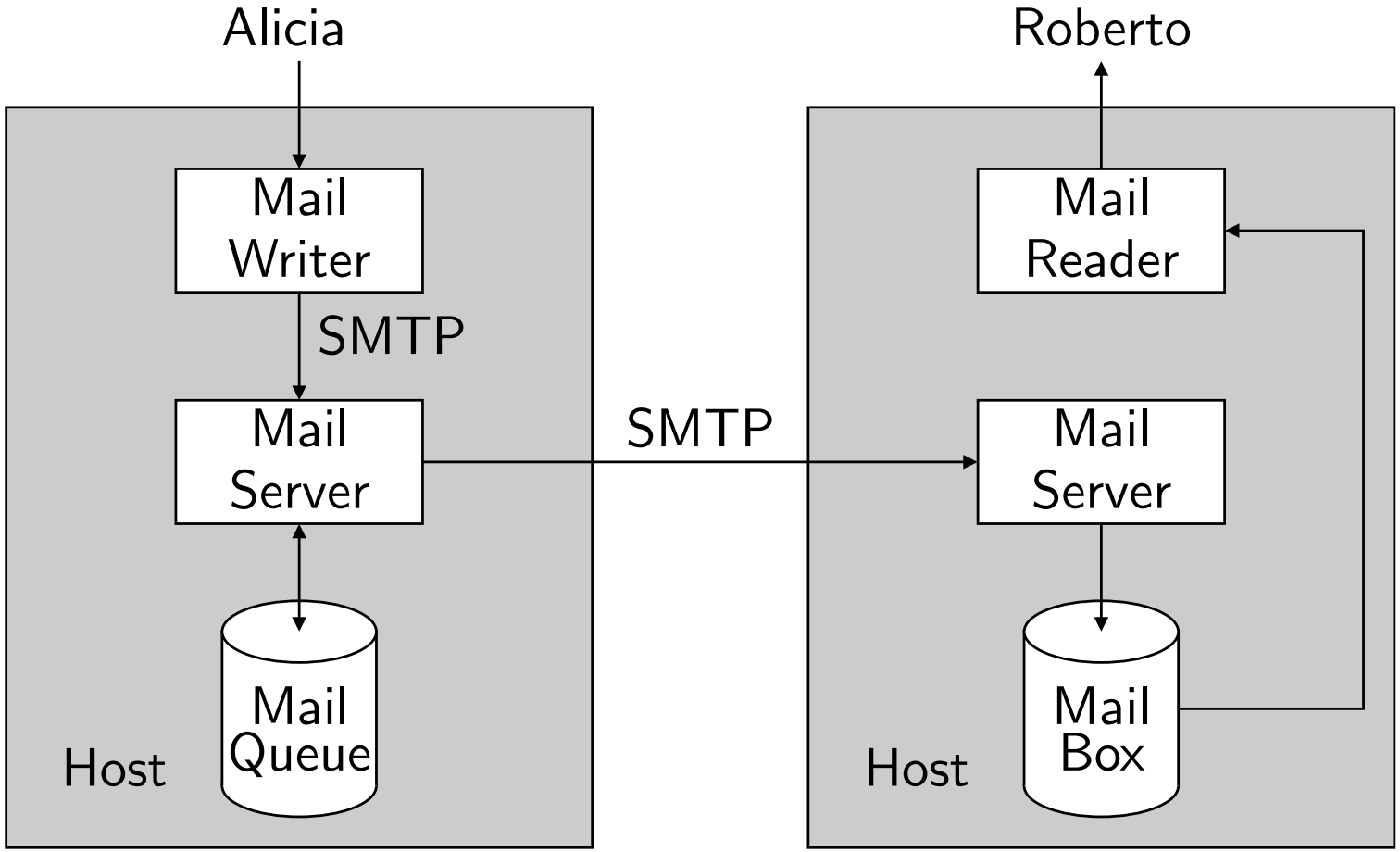
[1] En Unix, típicamente en `/var/spool/mail/Roberto`.

7.2.2. Correo local usando lectores y escritores de correo



- Los lectores y escritores facilitan la tarea de componer y leer e-mails.

7.2.3. Correo remoto usando servidores locales



- El servidor de Alicia tratará de enviar los e-mails inmediatamente. Si la comunicación con el servidor de Roberto no es posible, los e-mails

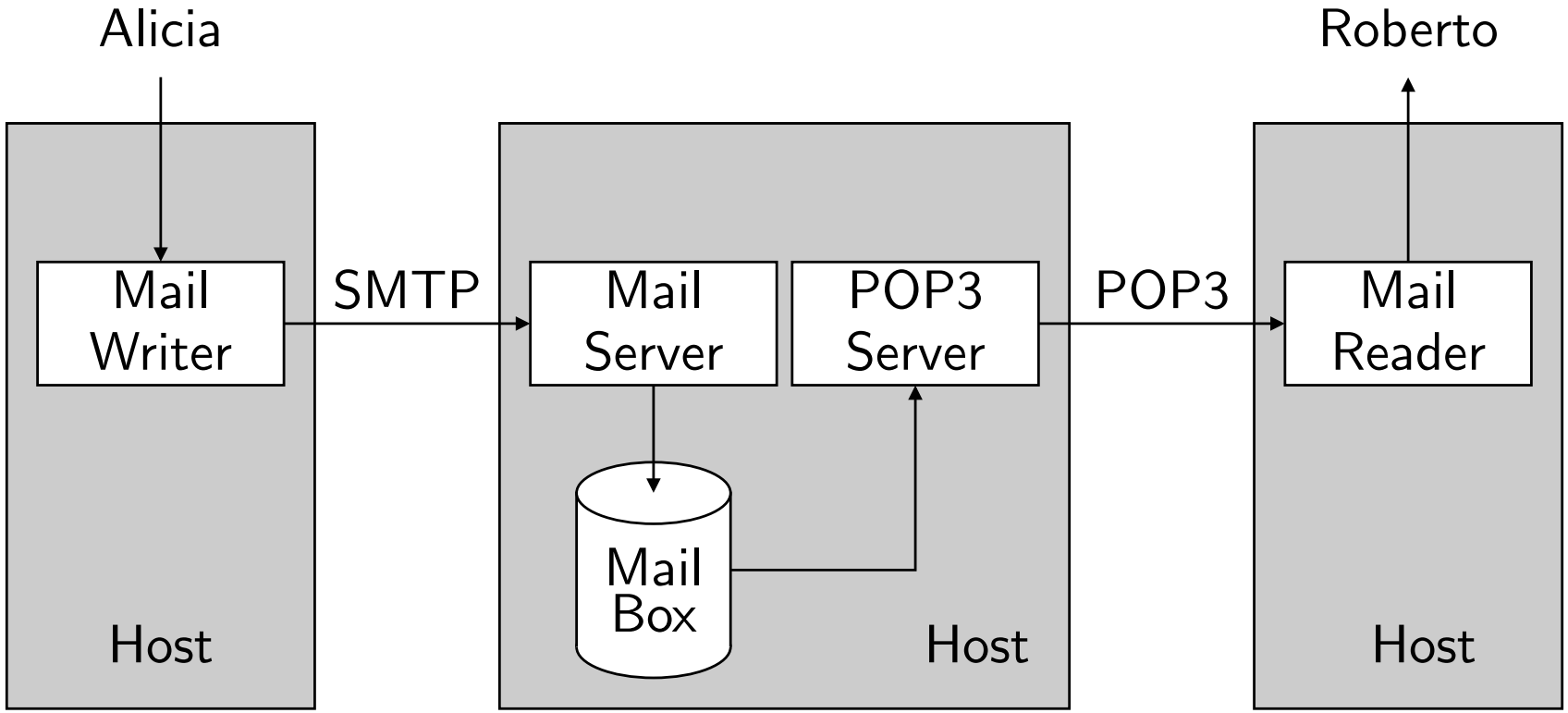
se almacenan en una cola local[2] para un intento posterior[3].

- Si después de varios días (normalmente 6) la transmisión no tiene éxito, entonces el servidor de correo envía un mensaje al usuario "escritor" indicándole que la comunicación no ha sido posible.

- Normalmente el número de saltos que realiza un e-mail entre servidores es sólo 1, es decir, el servidor de Alicia le envía el e-mail directamente al servidor de Roberto. Sin embargo, esto puede cambiar si el servidor de Alicia está configurado para que todo el correo saliente pase por otro servidor de correo intermedio. Estos servidores hacen "relay" (retransmisión) y se suelen utilizar para controlar si los mensajes contienen virus, etc.

[2] En Unix, típicamente en `/var/spool/mqueue/Alicia`.
[3] Que se realiza aproximadamente cada 30 minutos.

7.2.4. Correo remoto usando un servidor remoto



- El escritor de correo utiliza SMTP para el correo (saliente) y el lector de correo POP3 (o IMAP) para el correo entrante.

7.3. El SMTP

- RFC 2821.
- Controla cómo se realiza la transmisión de datos entre los servidores de correo.
- Es bastante antiguo[4] y sólo permite enviar mensajes escritos en ASCII de 7 bits.
- Corre sobre el TCP y se utiliza el puerto 25.
- El SMTP utiliza conexiones persistentes, lo que significa que si el emisor tiene que transmitir varios e-mail's, lo hará sobre la misma conexión TCP.

[4] 1982, teniendo en cuenta que Internet apareció formalmente a principios de los 70.

- Un ejemplo (servidor, cliente):

```
hundix$ telnet localhost 25
220 hundix.ace.ual.es ESMTP Sendmail 8.12.8/8.12.8;
Sun, 11 Jan 2004 10:31:23 +0100
HELO hundix.ace.ual.es
250 hundix.ace.ual.es Hello localhost.localdomain
[127.0.0.1], pleased to meet you
MAIL FROM: <vruiz@ual.es>
250 2.1.0 <vruiz@ual.es>... Sender ok
RCPT TO: <bill_gates@msn.com>
250 2.1.5 <bill_gates@msn.com>... Recipient ok (will queue)
DATA
354 Enter mail, end with "." on a line by itself
Como va tu Windows?
.
250 2.0.0 i0B9VNiA001623 Message accepted for delivery
QUIT
221 2.0.0 hundix.ace.ual.es closing connection
Connection closed by foreign host.
```

7.4. Formato de un e-mail

- Un e-mail (tal y como se almacena en una mailbox tras su recepción) se compone de una cabecera y de un cuerpo de mensaje, separados por una línea en blanco (CRLF).
  - El formato de la **cabecera** está estandarizado (RFC 822). Es una lista de líneas de la forma:

    palabra-clave: argumento

    Algunas de las palabras-clave (junto con sus argumentos) son obligatorias para que un e-mail esté bien formateado. Este es un ejemplo mínimo:

    ```
    From: vruiz@ual.es
    To: Bill_Gates@msn.com
    Subject: Has probado el Linux?
    ```

    Algunas entradas de la cabecera las coloca el escritor de correo (cuando lo construye) y otras son generadas por los servidores

de correo (cuando transmiten los e-mails). Algunos ejemplos:

`From:`	Escritor de correo
`To:`	Escritor de correo
`Subject:`	Escritor de correo
`Received:`	Servidor de correo y retransmisiores
`Return-Path:`	Retransmisores

- El **cuerpo del mensaje** no tiene formato (puede ser cualquier cosa siempre que se codifique en ASCII de 7 bits).

7.5. Las extensiones MIME

- RFC's 2045 y 2046.
- Las extensiones MIME (Multipurpose Internet Mail Extensions) amplian las posibilidades definidas en el RFC 822, para poder enviar cualquier cosa (caracteres con acentos, ficheros MP3, etc.) en el cuerpo del mensaje de un e-mail, utilizando códigos ASCII de 7 bits.
- Cuando transmitimos ficheros en un e-mail, la cabecera contiene las entradas:

```
Content-Transfer-Encoding: <método de codificación>
Content-Type: <tipo>/<subtipo>; <parámetros>
```

- Estas cabeceras son creadas por los escritores de correo.

Un ejemplo simple, en el que se transmite una imagen JPEG, sería:

```
From: alicia@crepes.fr
To: Roberto@hamburger.edu
Subject: Imagen de un delicioso crepe.
Content-Transfer-Encoding: base64
Content-Type: image/jpeg
980v98t0298t2098t2j0982098td0j928t9jtjd2j89t2j3
...
098309d20jr82j309h8239r209384fr239d8rj23098rd29
```

- Los tipos de mensajes MIME más usuales son:

  1. **text**: se utiliza para indicar que el contenido es texto, aunque en algún juego de caracteres especial. Ejemplo:

```
From: alicia@crepes.fr
To: roberto@hamburger.edu
Subject: Saludos
Content-Type: text/plain; charset="ISO-8859-1"
Hola Roberto.
Cómo estás?
```

2. **image**: indica que se envía una imagen. Ejemplo:

```
From: alicia@crepes.fr
To: Roberto@hamburger.edu
Subject: Imagen de un delicioso crepe.
Content-Transfer-Encoding: base64
Content-Type: image/gif
980v98t0298t2098t2j0982098td0j928t9jtjd2j89t2j3
...
098309d20jr82j309h8239r209384fr239d8rj23098rd29
```

3. **application**: se emplea para indicar que se transmite cualquier cosa que no sea texto ni una imagen. Ejemplo:

```
From: alicia@crepes.fr
To: roberto@hamburger.edu
Subject: El documento que me pediste
Content-Transfer-Encoding: base64
Content-Type: application/msword
8r9eq098re09q8g0hyr9egy908q0rehy9fqey90hrg9qhye
...
9809rj8390qd9j8e90jf8qw98jef9j8qwe980jq9j8freq9
```

4. **multipart**: se utiliza para enviar en un fichero e-mail varios ficheros adjuntos. Ejemplo:

```
From: alicia@crepes.fr
To: roberto@hamburger.edu
Subject: Imagen de un delicioso crepe con comentarios.
MIME-Version: 1.0
Content-Type: multipart/mixed; Boundary=StartOfNextPart
--StartOfNextPart
Content-Type: text/plain; charset="ISO-8859-1"
Querido Roberto,
te envío una imagen de un delicioso crepe.
--StartOfNextPart
Content-Transfer-Encoding: base64
Content-Type: image/jpeg
ldfljkasdfw983909298feq9hjfr939dr39r239j4rd29r3
...
9n4ytgfy9j8f98fepw98fk'fujweifuwe98rugf29tu924u
-- StartOfNextPart
Content-Type: text/plain; charset="ISO-8859-1"
Dime si quieres la receta.
```

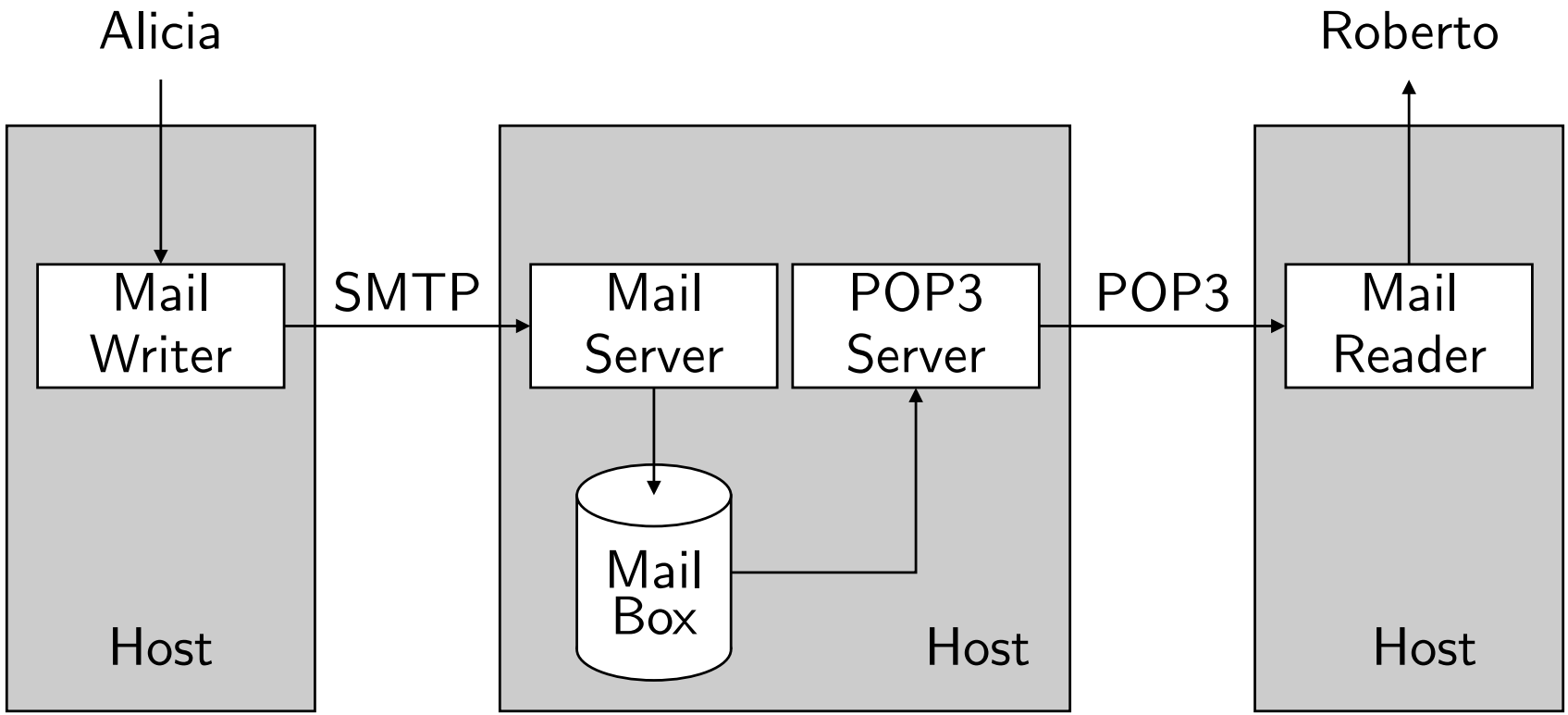
7.6. Los lectores/escritores de correo

- El correo electrónico sólo sería utilizado por los gurús de las redes si no existieran estos programas que ayudan a componer, revisar, ... e-mails. Sus principales funciones son:

  1. Gestionar las cabeceras.
  2. Comunicarse con un servidor de correo para enviar un e-mail usando SMTP.
  3. Avisar de que se ha recibido un nuevo e-mail.
  4. Gestionar la creación y presentación de mensajes con MIME.
  5. Gestionar los mailboxes (correo entrante).

7.7. Protocolos de acceso a correo (POP3 e IMAP)

- Sirven para que un usuario pueda gestionar (principalmente leer y borrar) su mailbox situado en un servidor remoto.



- Las ventajas más frecuentes por las que un usuario desea leer el correo en su host local (diferente del host que ejecuta el servidor de correo) son:

  1. El host local no tiene que estar siempre encendido; los e-mails se reclaman o envían al servidor de correo que no se apaga.
  2. La visualización de e-mails con contenidos multimedia (música, vídeo) es muy costosa (en términos de ancho de banda pico) si se hace mediante streaming (transmisión del audio y del vídeo a través de la red mientras se "consume").

- Los principales protocolos de acceso al correo son POP3 (Post Office Protocol – versión 3) – RFC1939 – e IMAP (Internet Mail Access Protocol) – RFC 2060 –. Ambos utilizan el TCP.

- El acceso se realiza en 3 fases:

  Fase 1. **Autentificación:** Al comienzo, el usuario se identifica (user y password).

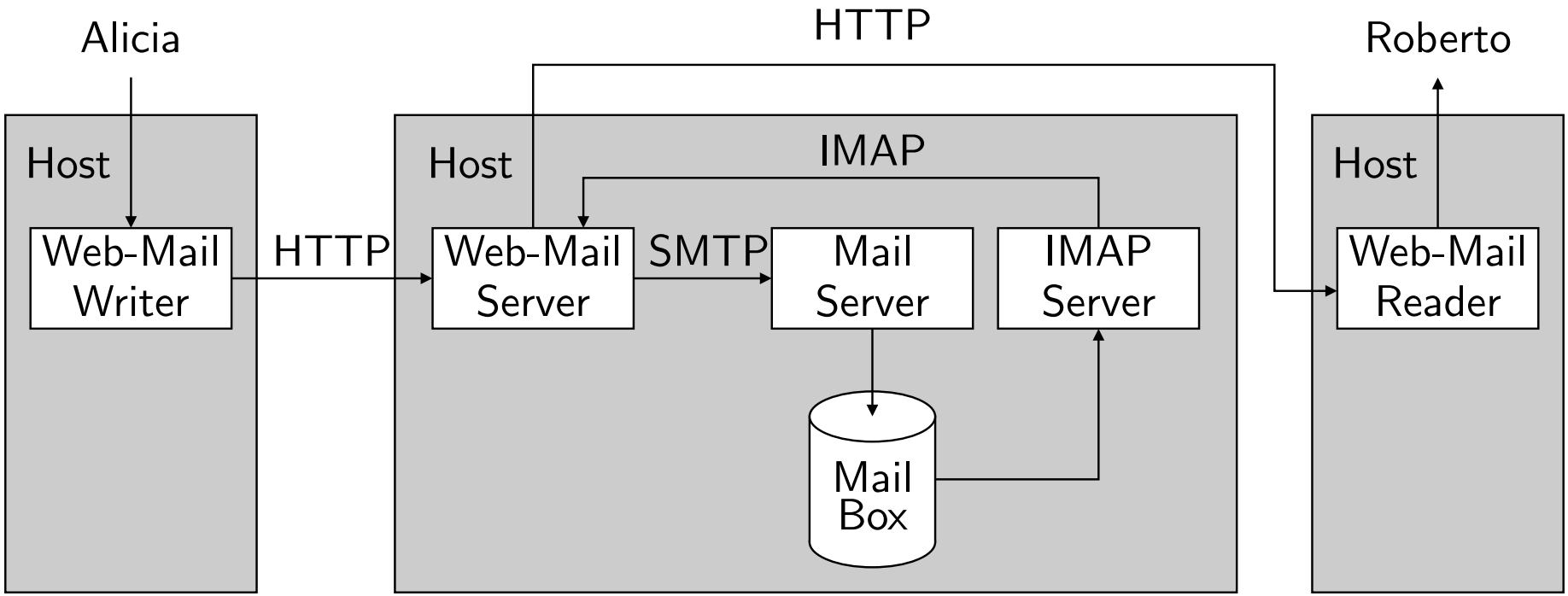
  Fase 2. **Transacción:** A continuación se transmiten los mensajes.

  - ◇ Con POP3 sólo se puede hacer dos cosas: (1) descargar y borrar o (2) sólo descargar.
  - ◇ Con IMAP se pueden crear carpetas en el servidor de forma que tras descargar un e-mail podemos eliminarlos del mailbox, pero dejándolo dentro de una carpeta. Con IMAP podemos también descargar únicamente un e-mail, o incluso una parte de un e-mail (esto es útil si hay e-mail tiene por ejemplo incluido (attached) un vídeo y la línea de conexión tiene muy poco de banda).

  Fase 3. **Actualización:** Durante esta etapa se cierra la conexión y se realiza el borrado/movimiento de los e-mails entre el mailbox y las carpetas.

7.8. Web-Based E-mail

- Para terminar de complicar el asunto, todavía existe una forma más de acceder al correo electrónico almacenado en un servidor: a través de la Web. Ejemplos son Hotmail y Yahoo Mail, aunque también existen servicios similares en universidades, empresas, ...
- El "agente de usuario" es un navegador Web y el servidor de acceso al correo es un servidor Web especialmente diseñado para este propósito. En este sentido, podemos decir que el HTTP es también un protocolo de acceso a correo, aunque la comunicación con el servidor se realiza mediante scripts que utilizan generalmente IMAP.
- La ventaja de usar Web-Based E-mail es evidente: no es necesario disponer de un agente de usuario, sólo de un navegador Web.


Alicia
HTTP
Roberto
Host
Host
IMAP
Host
Web-Mail
Writer
HTTP
Web-Mail
Server
SMTP
Mail
Server
IMAP
Server
Web-Mail
Reader
Mail
Box

Capítulo 8

El DNS (Domain Name Service)

8.1. Función del DNS

- En Internet cada host público se identifica mediante su dirección IP.
- Las dirs IP son números de 32 bits en IPv4 y de 128 bits en IPv6. En IPv4, la forma más corriente de especificarlas es:

  A.B.C.D

  donde A, B, C y D son números entre 0 y 255.
- Esto para los humanos no es muy cómodo porque nos cuesta mucho recordar números. Nosotros utilizamos mejor nombres de hosts.
- El DNS se encarga de traducir los nombres en sus correspondientes dirs IP [14].

8.2. Descripción del DNS

- RFC's 1034 y 1035.
- El DNS es una base de datos distribuida (ningún host servidor de DNS, también llamado *servidor de nombres*, conoce todas las traducciones).
- Los servidores de DNS son habitualmente máquinas Unix que ejecutan el demonio BIND (Berkeley Internet Name Domain).
- Los servidores de nombres escuchan a través del puerto 53.
- El DNS es un servicio crítico para el funcionamiento de Internet porque la mayoría de las aplicaciones que interactuan con humanos necesitan la resolución de nombres constantemente.
- El DNS se implementa sobre el UDP.

8.3. Alias y nombres canónicos

- Un host puede tener más de un nombre, uno canónico y el resto alias. Esto se hace normalmente para concentrar sevicios.

  Ejemplo 8.1: Podemos hacer que un mismo host sea servidor de correo electrónico y de la Web. El nombre canónico del host podría ser "X.Y.Z" y su alias "www.Y.Z". Por ejemplo, `filabres.ual.es` (el nombre canónico) y `www.ual.es` (el alias) fueron durante un tiempo la misma máquina `150.214.156.2`.

- Un mismo nombre (canónico o alias) puede ser asignado a muchos hosts. Esto se hace para:

  1. Descentralizar servicios (distribuir la carga de trabajo): Cuando nos conectamos a `cnn.com`, no lo hacemos siempre al mismo servidor:

```
$ /usr/bin/host cnn.com
cnn.com has address 64.236.24.12
cnn.com has address 64.236.24.20
cnn.com has address 64.236.24.28
cnn.com has address 64.236.16.20
cnn.com has address 64.236.16.52
cnn.com has address 64.236.16.84
cnn.com has address 64.236.16.116
cnn.com has address 64.236.24.4
```

2. Aumentar la resistencia a errores: Por ejemplo, cuando escribimos a un usuario de correo electrónico de `yahoo.es` en realidad escribimos a un servidor SMTP en uno de estos hosts:

```
$ /usr/bin/host -t MX yahoo.es
yahoo.es mail is handled by 1 mx2.mail.yahoo.com.
yahoo.es mail is handled by 5 mx4.mail.yahoo.com.
yahoo.es mail is handled by 1 mx1.mail.yahoo.com.
```

3. Que seamos atendidos por el servidor más cercano (ahorrar ancho de banda): Cuando nos conectamos a `www.google.com` desde un host en España somos automáticamente redirigidos a `www.google.es`[1]:

```
$ /bin/ping www.google.es
PING www.google.akadns.net (66.102.11.104) 56(84) bytes of data
64 bytes from 66.102.11.104: icmp_seq=1 ttl=234 time=94.1 ms
:
$ /bin/ping www.google.com
PING www.google.akadns.net (66.102.11.99) 56(84) bytes of data
64 bytes from 66.102.11.99: icmp_seq=1 ttl=235 time=94.1 ms
:
$ /usr/bin/host www.google.akadns.net
www.google.akadns.net has address 66.102.11.99
www.google.akadns.net has address 66.102.11.104
```

[1] Nótese que la carga de `www.google.akadns.net` está distribuida.

8.4. Arquitectura del DNS

- El DNS es un sistema cliente-servidor, donde la aplicación servidora está distribuida. Si esto no fuera así el sistema no escalaría porque:
  1. Existiría un único punto de fallo.
  2. Sería un cuello de botella.
  3. La base de datos con los registros DNS estaría alejada de la mayoría de los host de Internet.
  4. Una base de datos enorme, en continuo cambio, recaería en una sola máquina y su mantenimiento sería muy costoso.
- El tiempo de respuesta es ilimitado, aunque generalmente es pequeño (décimas de segundo). Esto tiene mucho que ver con que el DNS se base en el UDP.

- Los servidores de nombres se organizan jerárquicamente. Dependiendo del nivel en que se encuentran hablamos de tres tipos: (1) servidores autorizados (authoritative), (2) servidores de dominio del alto nivel (top-level domain) y (3) servidores raíz (root):

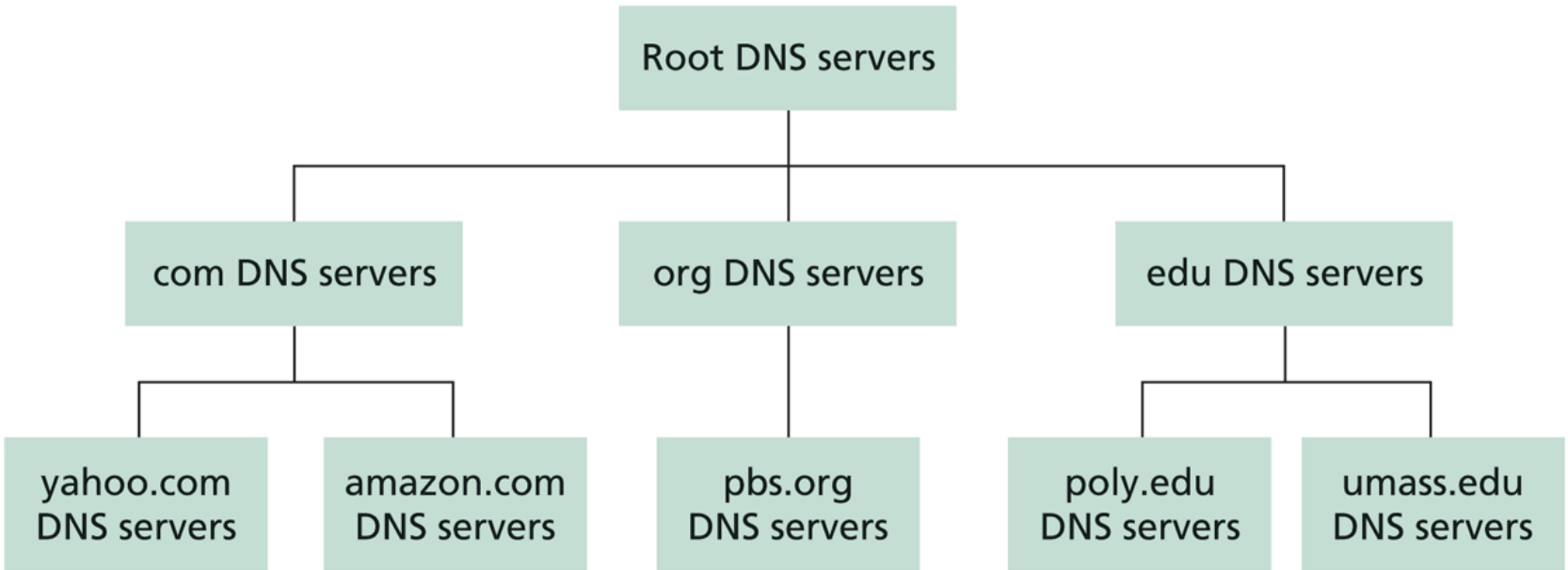


Figure 2.18 ♦ Portion of the hierarchy of DNS servers

Para aumentar la resistencia a errores, en cada dominio suelen existir varios servidores de nombres.

8.4.1. Servidores de nombres raíz

- Hasta la fecha de marzo del 2008, en Internet existen 13 servidores (generalmente muy potentes debido a la alta carga que deben soportar) de nombres raíz (`http://www.root-servers.org`).

Figure 2.19 ♦ DNS root servers in 2004 (name, organization, location)

8.4.2. Servidores de nombres de alto nivel

- Son responsables de los dominios de alto nivel: `com`, `org`, `edu`, `gov`, `es`, ...
- La definición del servidor de nombres de alto nivel (TLD) es un poco ambigüa porque la "altura" es una medida relativa con respecto a otros servidores de nombres subordinados.

  Ejemplo 8.2: Cada departamento de una universidad podría tener su propio servidor de nombres para los hosts de ese departamento. Por ejemplo, el departamento de arquitectura de computadores y electrónica de la universidad de Almería podría tener un servidor para el dominio `ace.ual.es`. Además, la universidad podría tener un servidor de nombres para el dominio `ual.es`. En este caso, el servidor de la universidad es un servidor de más alto nivel que él del departamento. Sin embargo, por encima al menos hay dos servidores de mayor nivel: el que atiende al dominio `es` y los servidores de nombres raíz.

8.4.3. Servidores de nombres autorizados

- Cada organización con hosts públicos (universidades, empresas, etc.) debe poseer al menos un servidor de nombres autorizado.
- Por definición, un servidor de nombres autorizado para el dominio `Y` debe de mantener los registros de resolución de todos los hosts `xxx.Y`.
- La fuente original de los registros de resolucion (también llamados registros de recursos) está en los servidores de nombres autorizados.

Ejemplo 8.3: Continuado con el ejemplo anterior, el servidor de nombres del departamento de arquitectura de computadores y electrónica sería el único servidor que debe mantener los registros de las máquinas del departamento. De hecho, cuando se instalase una nueva máquina, el proceso de alta de dicho host en el DNS consistiría en crear un nuevo registro sólo en dicho servidor.

8.5. Las consultas

- Cuando una aplicación solicita una resolución, primero consulta a uno de sus servidores de nombres locales (situados en el nivel más bajo del mismo dominio), que suele ser uno de los servidores de nombres autorizados para ese nivel del dominio (los servidores de nombres pueden estar replicados para aumentar la fiabilidad del DNS). Esto se hace así porque es muy probable que la mayoría de las resoluciones cacheadas que se repiten sean referentes a hosts del mismo dominio.

  Ejemplo 8.4: Cuando el host `gogh.ace.ual.es` solicita una resolución, este consulta a `filabres.ual.es` o a `alboran.ual.es`. `miro.ace.ual.es` (otro host del mismo dominio) hace igual:

```
gogh$ /usr/bin/host -v gogh.ace.ual.es
Trying "gogh.ace.ual.es"
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 24177
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
;gogh.ace.ual.es.                IN      A

;; ANSWER SECTION:
gogh.ace.ual.es.        172800  IN      A       193.147.118.57

;; AUTHORITY SECTION:
ace.ual.es.             172800  IN      NS      filabres.ual.es.
ace.ual.es.             172800  IN      NS      alboran.ual.es.

;; ADDITIONAL SECTION:
filabres.ual.es.        172800  IN      A       150.214.156.2
alboran.ual.es.         172800  IN      A       150.214.156.32

Received 126 bytes from 150.214.156.2#53 in 23 ms
```

```
gogh$ /usr/bin/host -v miro.ace.ual.es
Trying "miro.ace.ual.es"
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 24036
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
;miro.ace.ual.es.                IN      A

;; ANSWER SECTION:
miro.ace.ual.es.        172800  IN      A       193.147.118.63

;; AUTHORITY SECTION:
ace.ual.es.             172800  IN      NS      filabres.ual.es.
ace.ual.es.             172800  IN      NS      alboran.ual.es.

;; ADDITIONAL SECTION:
filabres.ual.es.        172800  IN      A       150.214.156.2
alboran.ual.es.         172800  IN      A       150.214.156.32

Received 126 bytes from 150.214.156.2#53 in 23 ms
```

- En otras ocasiones una aplicación solicita la resolución de un nombre que no pertenece a su dominio y por tanto un servidor local no puede traducirlo. Entonces el servidor local se comunica con un servidor de nombres de ámbito superior (que suele ser un servidor de nombres raíz, por sencillez para los administradores de las redes).

- Hay muchos casos donde el DNS está muy replicado por tratarse de un servico crítico. Por ejemplo, según la siguiente consulta existen 7 servidores de nombres que son autorizados para el dominio `ual.es`. En este ejemplo se muestra también el TTL (Time To Live) usado para el chacheo de los registros devueltos por ese servidor y el tipo de servidor (A = Authorized, NS = Name Server).

```
$ /usr/bin/host -v filabres.ual.es
Trying "filabres.ual.es"
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 36716
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 7, ADDITIONAL: 7

;; QUESTION SECTION:
;filabres.ual.es.               IN      A

;; ANSWER SECTION:
filabres.ual.es.        172800  IN      A       150.214.156.2

;; AUTHORITY SECTION:
ual.es.                 172800  IN      NS      filabres.ual.es.
ual.es.                 172800  IN      NS      alboran.ual.es.
ual.es.                 172800  IN      NS      dns1.cica.es.
ual.es.                 172800  IN      NS      dns2.cica.es.
ual.es.                 172800  IN      NS      sun.rediris.es.
ual.es.                 172800  IN      NS      chico.rediris.es.
ual.es.                 172800  IN      NS      ineco.nic.es.

;; ADDITIONAL SECTION:
filabres.ual.es.        172800  IN      A       150.214.156.2
alboran.ual.es.         172800  IN      A       150.214.156.32
dns1.cica.es.           17790   IN      A       150.214.5.83
dns2.cica.es.           14781   IN      A       150.214.4.35
sun.rediris.es.         19453   IN      A       130.206.1.2
chico.rediris.es.       19673   IN      A       130.206.1.3
ineco.nic.es.           2379    IN      A       194.69.254.2

Received 310 bytes from 150.214.156.2#53 in 33 ms
```

Ejemplo 8.5: Consultas recursivas. `cis.poly.edu` necesita obtener la dir IP de `gaia.cs.umass.edu`.

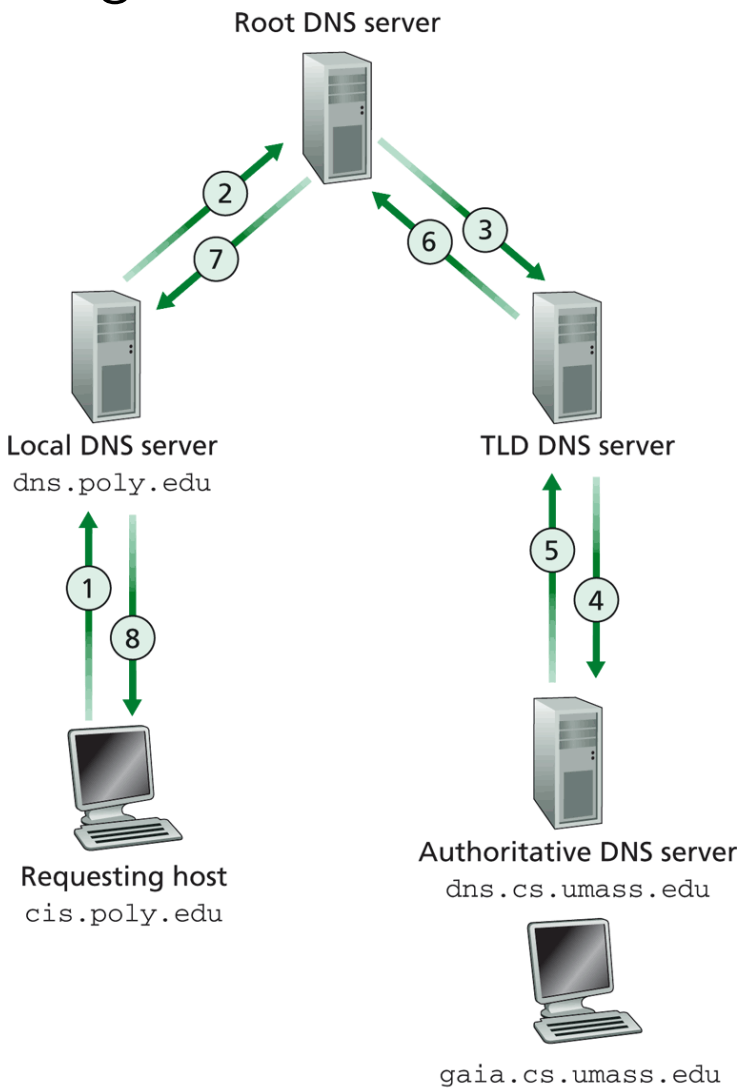


Figure 2.21 ♦ Recursive queries in DNS

Ejemplo 8.6: Consultas iterativas. `cis.poly.edu` necesita obtener la dir IP de `gaia.cs.umass.edu`.

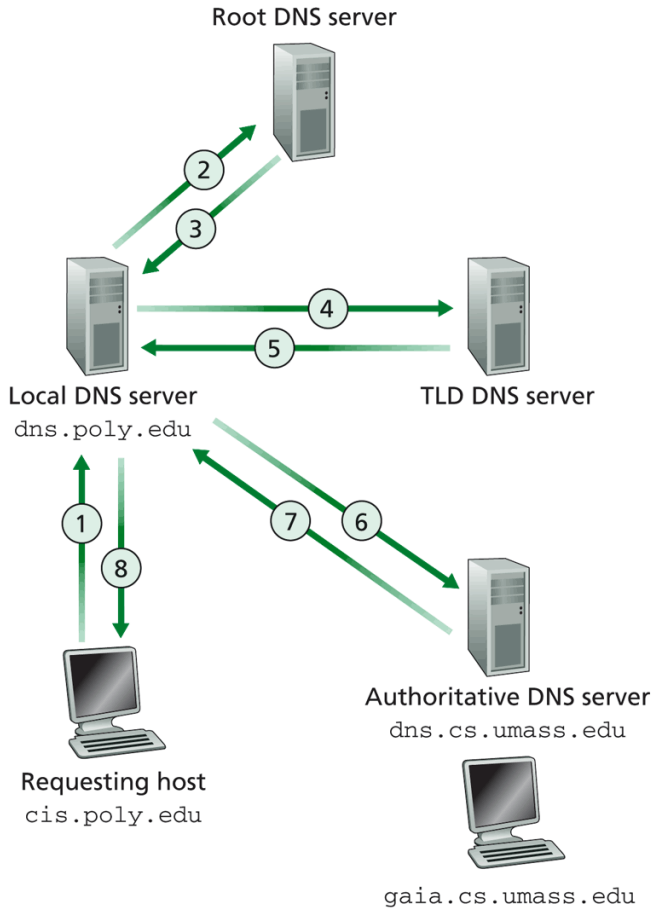


Figure 2.20 ♦ Interaction of the various DNS servers

8.6. Caching

- Cuando un servidor de nombres recibe una correspondencia DNS (una traducción), normalmente realiza una copia en su memoria local, con la esperanza de que en poco tiempo se le realize la misma petición.
- Si ésta se produce, el servidor de nombres resuelve, aunque en este caso no se trate del servidor autorizado. Como consecuencia el tiempo de respuesta se minimiza.
- Para poder tratar con hosts efímeros (que cambián constantemente de IP), los registros de las cachés son destruidos después de un periodo de tiempo (normalmente 2 días).

8.7. Los registros DNS

- Los registros DNS son intercambiados por los servidores de nombres para proporcionar el servicio de resolución.
- Los registros DNS tienen 4 campos:

  `(Nombre, Tipo, Valor, TTL)`

  El campo `TTL` es el tiempo de vida del registro e indica cuándo debe ser borrado de las cachés de los servidores. Los otros campos dependen del valor que toma el campo `Tipo`:

  1. Si `Tipo` == `A`, `Nombre` es un host y `Valor` es su dir. IP. Este tipo de registro se utiliza para una resolución estándar. Por ejemplo, cuando preguntamos por la dir IP de `dali.ace.ual.es` recibimos un registro de la forma:

     `(dali.ace.ual.es, 193.147.118.56, A, TTL)`

     Estos registros están de forma permantente en los servidores de nombres autorizados.

Ejemplo 8.7: Consultando la dir IP de www.ual.es:

```
$ /usr/bin/host -t A www.ual.es
www.ual.es has address 150.214.156.62
```

2. Si `Tipo` == `NS`, `Nombre` es un dominio y `Valor` es su servidor de nombres autorizado. Por ejemplo, cuando queremos preguntar por el servidor de nombres autorizado para el dominio `ual.es` recibimos:

   `(ual.es, dns.ual.es, NS, TTL)`

   Ejemplo 8.8: Consultando el/los servidor/es de nombre/s de `gogh.ace.ual.es`: (ver Sección 8.5)

3. Si `Tipo` == `CNAME`, `Nombre` es un alias y `Valor` es su nombre canónico. Por ejemplo, cuando queremos conocer el nombre canónico del alias `www.ace.ual.es` recibimos:

   `(www.ace.ual.es, dali.ace.ual.es, CNAME, TLL)`

   Ejemplo 8.9: Consulando el nombre canónico de `www.ace.ual.es`:

```
$ /usr/bin/host -t CNAME www.ace.ual.es
www.ace.ual.es is an alias for dali.ace.ual.es.
```

4. Si `Tipo` == `MX`, `Nombre` es un dominio de correo y `Valor` es el nombre canónico del servidor de correo. Por ejemplo, cuando utilizamos el alias de correo `ual.es` en realidad mandamos el e-mail a `smtp.ual.es` porque el DNS constesta con:

   `(ual.es, smtp.ual.es, MX, TLL)`

   Nótese que una compañía puede tener el mismo alias par su servidor de correo y para su servidor Web. Cuando estamos utilizando el correo electrónico, el cliente DNS consultaría un registro DNS con `Tipo` == `MX`. Sin embargo, para obtener el nombre canónico del servidor Web, consultaría con `Tipo` == `CNAME`.

   Ejemplo 8.10: Consultando por el servidor de SMTP de la UAL:

```
$ /usr/bin/host -t MX ual.es
ual.es mail is handled by 10 mail.rediris.es.
ual.es mail is handled by 2 smtp.ual.es.
```

8.8. Regional Internet Registries

- Controlan la asignación de direcciones IP y los nombres de dominio de alto nivel.
- Actualmente existen 5 organismos de registro:
  1. American Registry for Internet Numbers (ARIN) (http://www.arin.net) para América del Norte.
  2. Réseaux IP Européens Network Coordination Centre (RIPE NCC) (http://www.ripe.net) para Europa, cercano Oriente y Asia central.
  3. Asia-Pacific Network Information Centre (APNIC) (http://www.apnic.net) para el resto de Asia y las regiones del Pacífico.
  4. Latin American and Caribbean Internet Address Registry (LACNIC) (http://www.lacnic.net) para América Latina y la región del Caribe.

5. African Network Information Centre (AfriNIC) (http://www.afrinic.net/) para Africa.

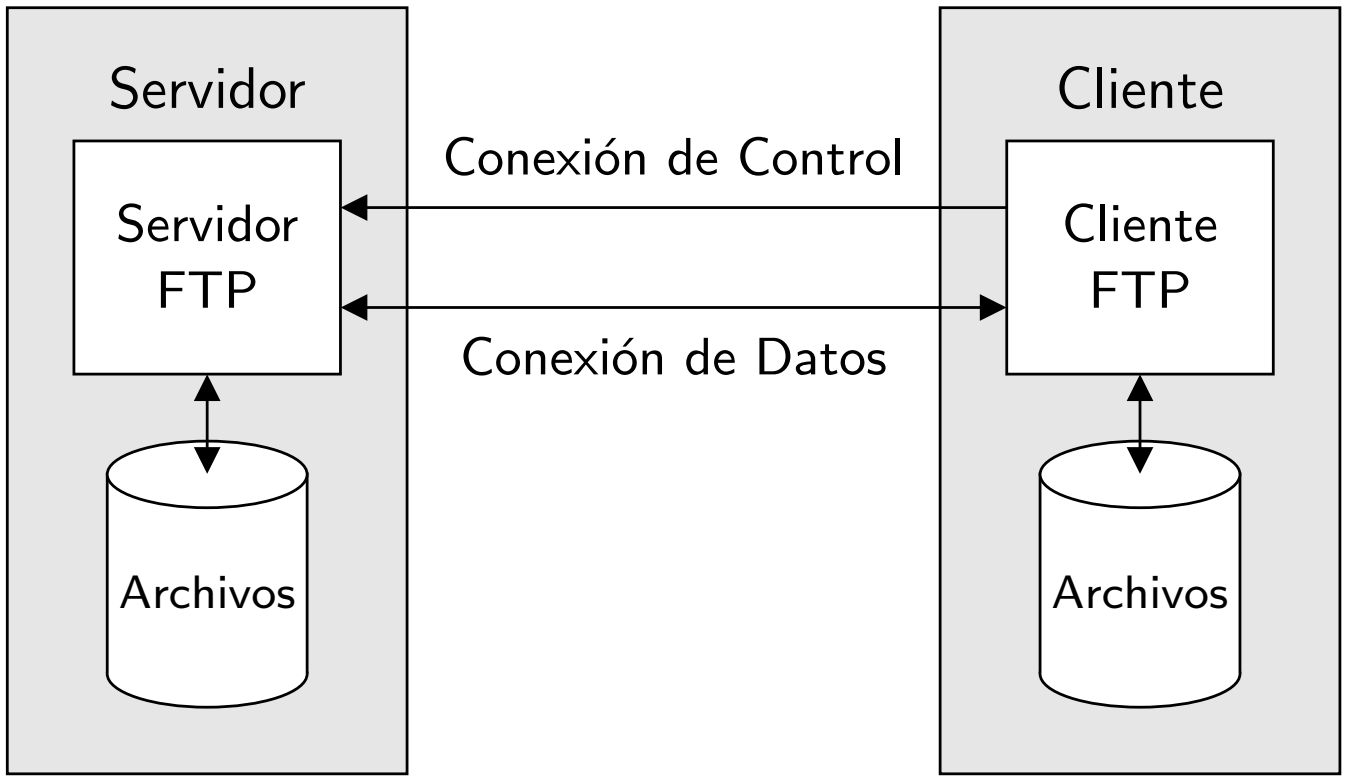
- Estos organismos están coordinados por la Internet Corporation for Assigned Names and Numbers (ICANN) (http://www.icann.org).

Capítulo 9

Compartición de Ficheros (File Sharing)

9.1. El FTP (File Transfer Program)

- La aplicación FTP permite la transmisión de archivos entre hosts remotos.



- Utiliza TCP en ambas conexiones.

- La conexión de control transporta la información (en formato ASCII) necesaria para controlar la transmisión de archivos (login/password, comandos para movernos por el sistema de ficheros remoto y mover archivos, listar directorios, ...). El servidor recibe la información de control a través del puerto 21 y utiliza el File Transfer Protocol (FTP, descrito en el RFC 959).
- La conexión de datos se utiliza para transferir 1 archivo (se establece una conexión TCP para cada archivo). El servidor utiliza para esto el puerto 20.

9.2. Aplicaciones P2P (Peer-to-peer)

- Muy alta escalabilidad (los clientes son también servidores).
- Se utilizan para compartir archivos que residen en los hosts de los usuarios. Por esto a las aplicaciones P2P también se les llama aplicaciones de compartición de archivos entre iguales. Ejemplos: Napster, Gnutella, KaZaA, eMule y BitTorrent.
- Contexto complejo: típicamente los hosts no permanecen todo el tiempo encendidos (ni conectados) y rara vez disponen de una dirección IP fija.
- Cuando un usuario busca un archivo obtiene una lista de hosts que en este momento están conectados y almacenan (al menos parcialmente) dicho archivo. Con esto se calcula un índice de accesibilidad al fichero.

- La transmisión (generalmente mediante TCP) de un archivo se produce directamente (sin servidores intermedios) entre uno o varios de esos hosts y el del usuario (que ha realizado la búsqueda). Esto permite transmitir simultáneamente distintas parte del mismo archivo mediante múltiples conexiones.

- En cuanto el usuario dispone de una sección del archivo, automáticamente entra a formar parte de la lista de iguales que disponen de él. Así, mientras lo descarga, otros usuarios pueden comenzar a descargarlo de él.

- Por tanto, las aplicaciones P2P de compartición de archivos son altamente escalables en lo que se refiere a la cantidad de datos que pueden gestionar, porque aunque un nuevo usuario consume ancho de banda de salida de los iguales que contienen los archivos que se está descargando, también proporciona el suyo para que otros hagan lo mismo.

- Sin embargo, en general los sistemas P2P presentan un problema de cuello de botella en los buscadores de contenidos. A continuación discutiremos las 3 estrategias existentes.

9.2.1. Búsqueda usando un directorio centralizado

- Ejemplos: Napster, eMule y BitTorrent.
- Existe un gran servidor (Napster) o un conjunto de servidores (parcialmente) replicados (eMule) para que proporcionan servicio de búsqueda. En el caso de BitTorrent, cada fichero `.torrent` especifica un servidor que no es de búsqueda de contenidos, sino de determinación de peers para ese contenido.
- Cada igual (cuando se conecta) informa al servidor de su dirección IP y de los contenidos que desea compartir. Así, el servidor crea y mantiene una base de datos dinámica que relaciona cada nombre de fichero con el conjunto de IP's de los hosts que lo contienen.

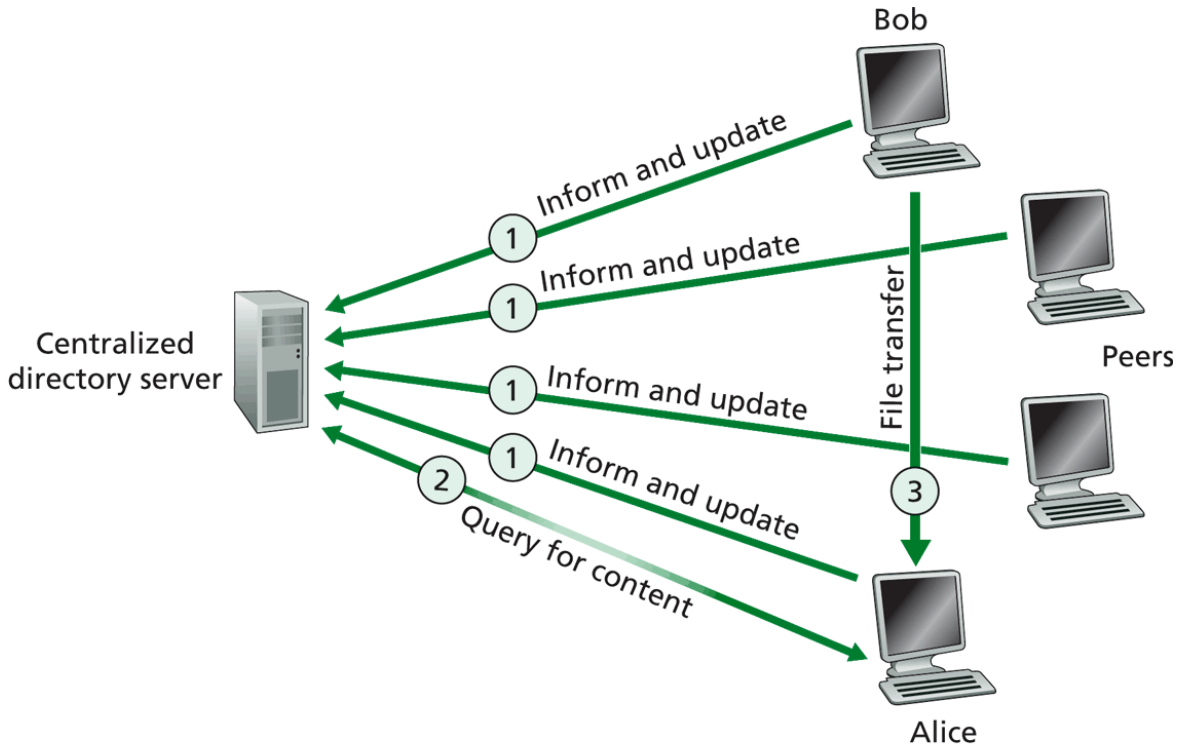


Figure 2.23 ♦ The P2P paradigm with a centralized directory

- Ventajas: la búsqueda de contenidos (por parte de los iguales) es sencilla porque la base de datos está centralizada.
- Desventajas: el servidor central se convierte en el cuello de botella del sistema y si no existe ninguno funcionando, ningún igual puede buscar.

9.2.2. Búsqueda usando un directorio descentralizado

- Ejemplos: KaZaA, Overnet, eDonkey y Kademlia (eMule).
- Un subgrupo de iguales son designados como líderes de grupo (supernodos) y cuando un nuevo igual se conecta, se le asigna uno de ellos.

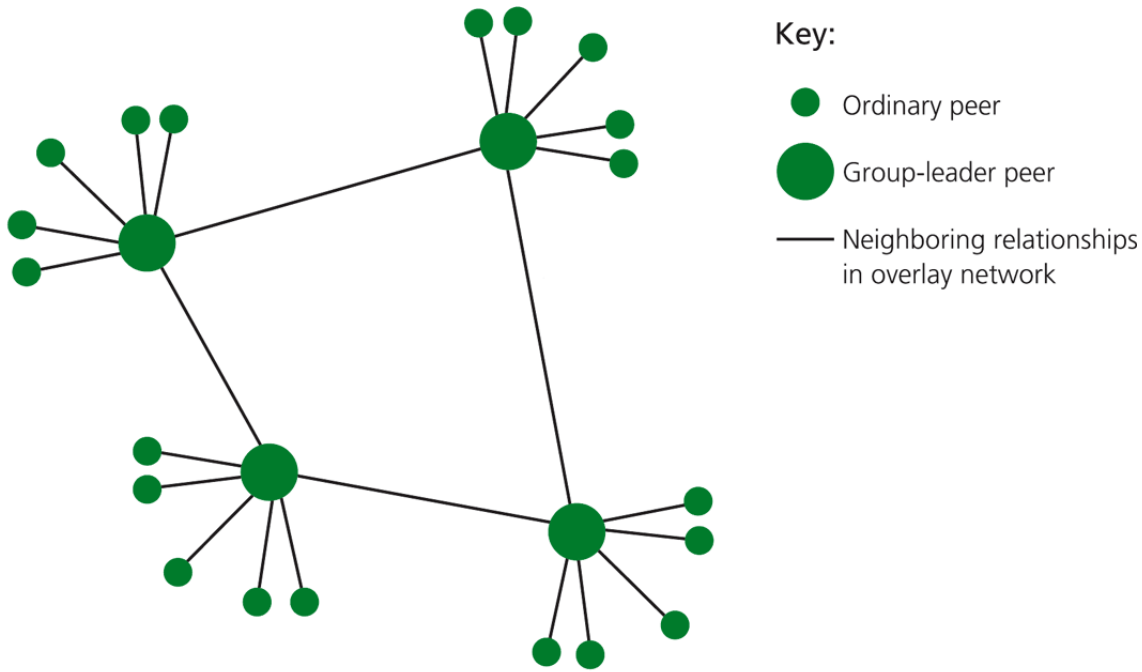


Figure 2.25 ♦ Hierarchical overlay network for P2P file sharing

- Cada líder de grupo gestiona una base de datos con la información de los iguales de su grupo.
- Cuando un usuario busca, lo hace primero dentro de su grupo y obtiene una contestación rápida. Si el usuario lo solicita, puede ampliar la búsqueda al resto de grupos ya que el líder de su grupo conoce a los demás líderes.
- Cuando un usuario se conecta lo hace a partir de un nodo de arranque que conoce al menos uno de los líderes de grupo. No es necesario conocer, de entrada, a un supernodo.
- Un nodo (un igual) se convierte en un líder normalmente porque el usuario lo solicita[1].

[1] Probablemente a cambio de gastar ancho de banda extra en ser servidor de consultas, obtenga mayores prioridades a la hora de acceder a los ficheros compartidos.

- Ventajas de la búsqueda descentralizada:
  - Las bases de datos son localmente más pequeñas (un líder sólo se encarga de un subconjunto de los iguales, generalmente los más próximos (menos hops)).
  - El sistema es más escalable y resistente a fallos porque ahora la base de datos global está distribuida.
- Desventajas:
  - Cuando un líder se apaga, los iguales tienen que buscar otro líder vecino y regenerar la base de datos local.
  - Los nodos no son todos iguales (los líderes, que también funcionan como igual, consumen más recursos que los iguales).
  - Sigue existiendo la necesidad de un nodo inicial de arranque que conozca al menos un líder y que no puede apagarse.

9.2.3. Búsqueda mediante inundación

- Ejemplo: Gnutella.
- No existe una jerarquía de servidores (supernodos) e iguales. Todos los participantes son iguales.
- Para que un nodo se conecte debe conocer, al menos, la dirección IP de un nodo que ya forme parte de la red de compartición.
- Las búsquedas se realizan mediante *inundación de consultas*:

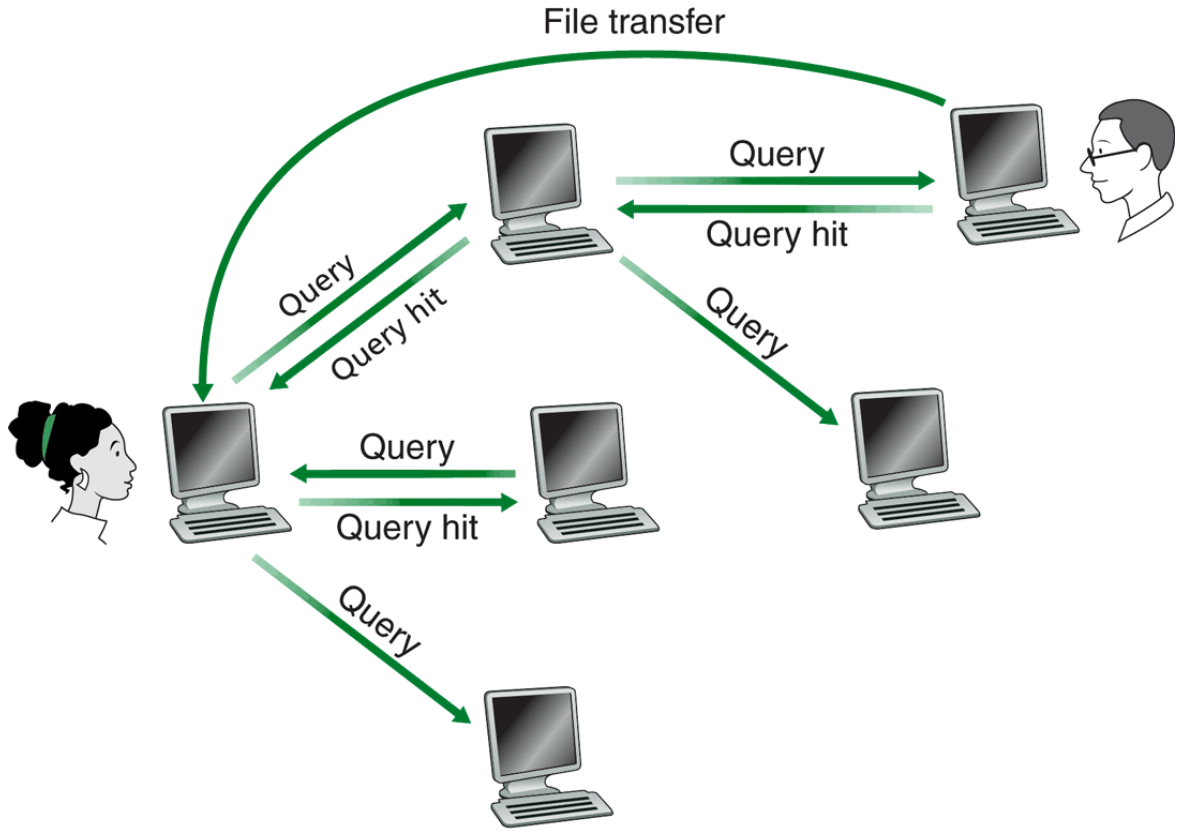


Figure 2.24 ♦ Search and file transfer in Gnutella

1. El nodo que inicia la búsqueda pregunta primero a sus vecinos (inmediatos).
2. Los vecinos repiten el proceso con sus vecinos inmediatos (excepto con los que le han preguntado ya por lo mismo).

3. Los nodos que contienen el archivo buscado se lo comunican al nodo vecino que le ha preguntado la búsqueda. Este proceso acaba cuando las contestaciones llegan hasta el nodo que inició la búsqueda.

- Ventajas:
  - Todos los nodos son iguales.
  - No existen bases de datos con información de directorio.
- Desventajas:
  - El volumen de tráfico generado por las consultas suele ser muy alto. Por este motivo el área de búsqueda (número de hops) suele estar limitado. Lo malo de esto es que puede ocurrir que sólo se obtengan búsquedas parciales.
  - Es necesario conocer un nodo de la red de compartición para poder formar parte de ella (`http://www.gnutella.com`).

9.3. Acerca de la tasa de descarga

- La tasa de descarga de las aplicaciones P2P depende normalmente de la tasa de subida a la red. Si ésta es alta, la de descarga también.
- Existen dos formas distintas de premiar al que más envía:
  1. Incrementando la prioridad del peer en las colas asociadas a los contenidos: "Si subes más, tardas menos tiempo de obtener conexiones". Esto es lo que ocurre en eMule, por ejemplo.
  2. Conectándote a los mejores peers. Hay protocolos, como BitTorrent, que el número de peers a los que te conectas está limitado (5 generalmente). Si los peers tratan de enviar datos a aquellos otros peers que mejor le sirven, aquellos peers que más ancho de banda dedican se comunicarán entre sí. De esta manera, "Si subes más, te conectarás a los mejores peers y por tanto, obtendrás más datos de la red". En BitTorrent cada 30 segundos la peor de las 5 conexiones se cierra y se establece una nueva (tras consultar al tracker) con la idea de mejorar algunas de las otras 4 conexiones existentes.

- En la práctica no debe dedicarse el 100 % del ancho de banda de subida porque ahogaríamos las conexiones TCP de descarga. Técnicamente lo que ocurre es que los segmentos de control de flujo que se envían a los peers se retrasan, provocándose un sobre-control del flujo.

9.4. Network File System

9.4.1. Características

- Desarrollado en 1984 por Sun Microsystems para SunOS. Más tarde donado al *public domain* [9].
- Permite montar un sistema de ficheros remoto dentro del sistema de ficheros local.[2] Las aplicaciones no ven la diferencia.
- El sistema de ficheros es "exportado" por el host remoto.
- El sistema de ficheros es "montado" por el host local.
- Muchos hosts remotos pueden montar el mismo sistema de ficheros y compartir así datos de forma sencilla.

[2] Un sistema de ficheros es cualquier directorio que cuelgue del directorio raíz en el host remoto (un directorio, una partición de un disco duro, un CD-ROM, un disco RAM, etc.).

- Las aplicaciones locales acceden al sistema de ficheros remoto como si fuera local. En otras palabras, el NFS es completamente transparente al usuario.
- Arquitectura cliente/servidor. El servidor se monta en el host que exporta el sistema de ficheros y el cliente, en los hosts que lo montan.
- El servidor asegura la integridad de los datos en el sistema de ficheros exportado.
- El NFS utiliza un protocolo a nivel de la capa de aplicación y se monta encima del sistema RPC, otro paquete de la capa de aplicación [25]. Dicho protocolo es el NFSP (NFS Protocol). Actualmente cohexisten tres versiones: la 2 [28], la 3 [29] y la 4 [31].
- Los datos son transmitidos a través de la red usando el sistema de representación XDR (eXternal Data Representation) [30]. Esto permite que hosts de diferentes clases (con distintos SO's, endian's y/o longitudes de palabra) puedan intercambiar datos.
-

9.4.2. El NFSP

- Es un protocolo sin estado lo que significa que ante un fallo del sistema (por ejemplo, que el servidor NFS se cuelgue), es el cliente el que sabe cómo recuperarse de los errores (por ejemplo, re-ejecutando la orden de escribir un fichero que no pudo ser escrito por el cuelgue del servidor).
- El protocolo se describe mediante un conjunto de procedimientos que se ejecutan sobre el sistema RPC. A continuación se muestran algunos de los más usados:

  1. `null() returns()`: Hace un ping al server. Sirve para medir latencias en la red.
  2. `lookup(dir_fh, name) return (fh, attr)`: Busca el fichero `name` en el directorio `dir_fh` y si lo encuentra, devuelve el descriptor de ese fichero `fh` más la información `attr` sobre los atributos del fichero.
  3. `create(dir_fh, name, attr) return(new_fh, new_attr)`: Crea en e directorio `dir_fh` el fichero `name`

con atributos attr. Si la creación tiene éxito, devuelve un nuevo descriptor de fichero new_fh que tiene los atributos new_attr.

4. remove(dir_fh, name) returns(status): Borra el fichero name en el directorio dir_fh y devuelve verdadero si la operación ha tenido éxito.
5. getattr(fh) returns(attr): Devuelve los atributos del fichero.
6. setattr(fh, attr) returns(new_attr): Establece los atributos del fichero.
7. read(fh, offset, count) returns(attr, data): Escribe en el puntero data los count bytes a partir del byte de índice offset del fichero fh. Se devuelve además los atributos del fichero.
8. write(fh, offset, count, data) returns(attr): Semejante al anterior método, excepto que escribimos en el fichero remoto.
9. rename(dir_fh, name, to_fh, to_name)

`returns(status)`: Renombra el fichero `name` en el directorio `dir_fh` al fichero `to_name` en el directorio `to_fh`. Se devuelve información sobre el éxito de la operación.

10. `link(dir_fh, name, to_fh, to_name) returns(status)`: Crea el fichero `to_name` en el directorio `to_fh`, que es un enlace al fichero `name` en el directorio `dir_fh`.[3]
11. `symlink(dir_fh, name, string) returns(status)`: Crea un enlace simbólico llamado `name` en el directorio `dir_fh` con el valor `string`.[4]
12. `readlink(fh) return(string)`: Devuelve el nombre del enlace simbólico.
13. `mkdir(dir_fh, name, attr) returns(fh, new_attr)`: Crea un nuevo directorio llamado `name` dentro del directorio

[3] Un enlace a un fichero es una nueva entrada en el sistema de ficheros que apunta a un fichero ya existente. Es como un seudónimo.

[4] La única diferencia entre un enlace (a secas, también llamado enlace duro) y un enlace simbólico es que éste último puede hacerse entre sistema de ficheros diferentes.

dir_fh, retorna el descriptor de ese directorio fh y los atributos new_attr con los que finalmente ha sido creado.

14. rmdir(dir_fh, name) returns(status): Elimina un directorio vacío llamado name que está dentro del directorio dir_fh y devuelve el resultado de dicha operación.

15. readdir(dir_fh, cookie, count) return(entries): En entries retorna hasta count bytes de las entradas en el directorio dir_fh. Cada entrada contiene un nombre de fichero, un identitificador y un puntero cookie a la siguiente entrada dentro del directorio. Este fichero permite en sucesivas llamadas ir recorriendo todas las entradas del directorio. Si cookie == NULL se devuelve la primera entrada.

16. statfs(fh) return(fs_stats): Devuelve información sobre el sistema de ficheros remoto como el tamaño de bloque, el número de bloques libres, etc.

Capítulo 10

Transmisión de audio y vídeo

10.1. Características de la transmisión de audio y vídeo

- Las aplicaciones que operan con secuencias de audio y vídeo son altamente sensibles al retraso de extremo-a-extremo (end-to-end delay) de la red y a la variación de este retraso (jitter) [14].
- Relacionado con lo anterior, generalmente necesitan una tasa de transmisión promedio sostenida porque una vez que la secuencia comienza a ser reproducida no pueden ocurrir "cortes" por culpa de la red.
- Normalmente toleran cierta cantidad de pérdida de datos durante la transmisión.
- En la mayoría de las transmisiones unicast se utiliza el UDP en lugar del TCP debido no hace falta controlar ni el flujo ni la congestión.
- Si la transmisión es multicast sólo puede utilizarse el UDP.

10.2. Ejemplos de aplicaciones

	Stored Media Transmission	Live Media Transmission	Real-time Media Transmission
Example(s):	Video on Demand	Internet TV, Internet Radio	Video-conferencing, IP telephony
Data-flow control:	Pause, rewind, fast-forward and indexing	None	None
Pre-fetching:	Yes, if extra band-width is available	No way	No way
Buffering:	As large as possible	Jitter hiding (few seconds)	< 500 ms
Internet transport protocol(s):	UDP/TCP	UDP	UDP

10.3. Obstáculos de la Internet actual

- **Tasas de transmisión variables:** las mayoría de los algoritmos de compresión utilizados para codificar audio y vídeo necesitan tasas de bits constantes.
- **Latencias altas y variables:** en muchos casos el retraso de extremo-a-extremo es demasiado alto (especialmente cuando los paquetes atraviesan enlaces congestionados) y además el jitter de los paquetes es distinto de cero.
- **No existen políticas de prioridad:** todos los paquetes son tratados por igual por los routers, por tanto, paquetes que no son sensibles al tiempo compiten por los mismos recursos que paquetes que sí lo son.
- **No existe el concepto de stream:** los paquetes pertenecientes al mismo stream de audio o de vídeo pueden desordenarse en la red.

10.4. ¿Cómo debería evolucionar Internet?

- A la hora de acomodar el tráfico sensible al tiempo, actualmente existen dos políticas radicalmente opuestas:

  1. **Permitir que las aplicaciones permitan reservar ancho de banda:** Esto significa fundamentalmente que:
     - Habría que modificar las políticas de servicio de los paquetes en los routers, lo que implica un gran cambio en su diseño actual.
     - Habría que idear alguna forma de controlar cúanto y cuándo se reservan los recursos (pagando por ello, por supuesto).
     - Habría que etiquetar los paquetes con información sensible al tiempo para que los routers los distinguieran.
  2. **Incrementar el ancho de banda y distribuir adecuadamente los contenidos:**
     - Los incrementos en las necesidades de ancho de banda se deben principalmente a que más gente se conecta a la red y

tiene accesos de mayor velocidad a través de su ISP. Es lógico esperar que dicho ISP gane dinero dando dicho servicio y que reinvierta parte de sus beneficios en infraestructura. En definitiva, la red va a crecer al mismo ritmo que se demandan recursos.

- Existen técnicas de replicación de contenidos que pueden ayudar a disminuir enormemente el tráfico. Por ejemplo, los ISP's deberían instalar en sus servidores aquellas secuencia de audio y vídeo que sus usuarios reclaman con mayor frecuencia.
- El concepto anterior debería además extenderse entre los distintos niveles de ISP formando lo que se conoce como una **red de superposición multicast**[1] (multicast overlay network).

[1] Es importante darse cuenta de que esta forma de realizar multicasting no tienen nada que ver con la que se realiza a nivel de IP. En IP del multicasting se realiza a nivel de red y en este caso el multicasting se hace a nivel de aplicación.

10.5. Problemas y soluciones en la transmisión de audio y vídeo

10.5.1. Eliminación del jitter

- ¿Por qué se retrasan los paquetes?
  - Aunque el emisor generalmente coloca los paquetes periódicamente en la red, los routers puede introducir un retardo variable en ellos dependiendo del estado de sus colas.
- ¿Cómo solucionamos el problema?
  - Retrasando la reproducción un tiempo prudencial (la máxima latencia de extremo-a-extremo esperada o bien un retraso que genere una tasa de pérdida de paquetes admisible) o hasta donde sea posible (a lo sumo 500 ms en el caso de las transmisiones en tiempo real).

10.5.2. Recuperación de paquetes perdidos

- ¿Por qué se pierden los paquetes?
  - Aunque el contenido de un paquete recibido puede ser diferente del enviado (lo que también puede ser un problema), lo normal es que los paquetes lleguen (sin errores) o no lleguen debido a los problemas de congestión de la red.
  - En otras ocasiones, aunque el paquete llega al destino lo puede hacer demasiado tarde, especialmente en el caso de la transmisión interactiva en tiempo real. En este caso el paquete se descarta en el receptor.

- ¿Cómo solucionamos el problema?

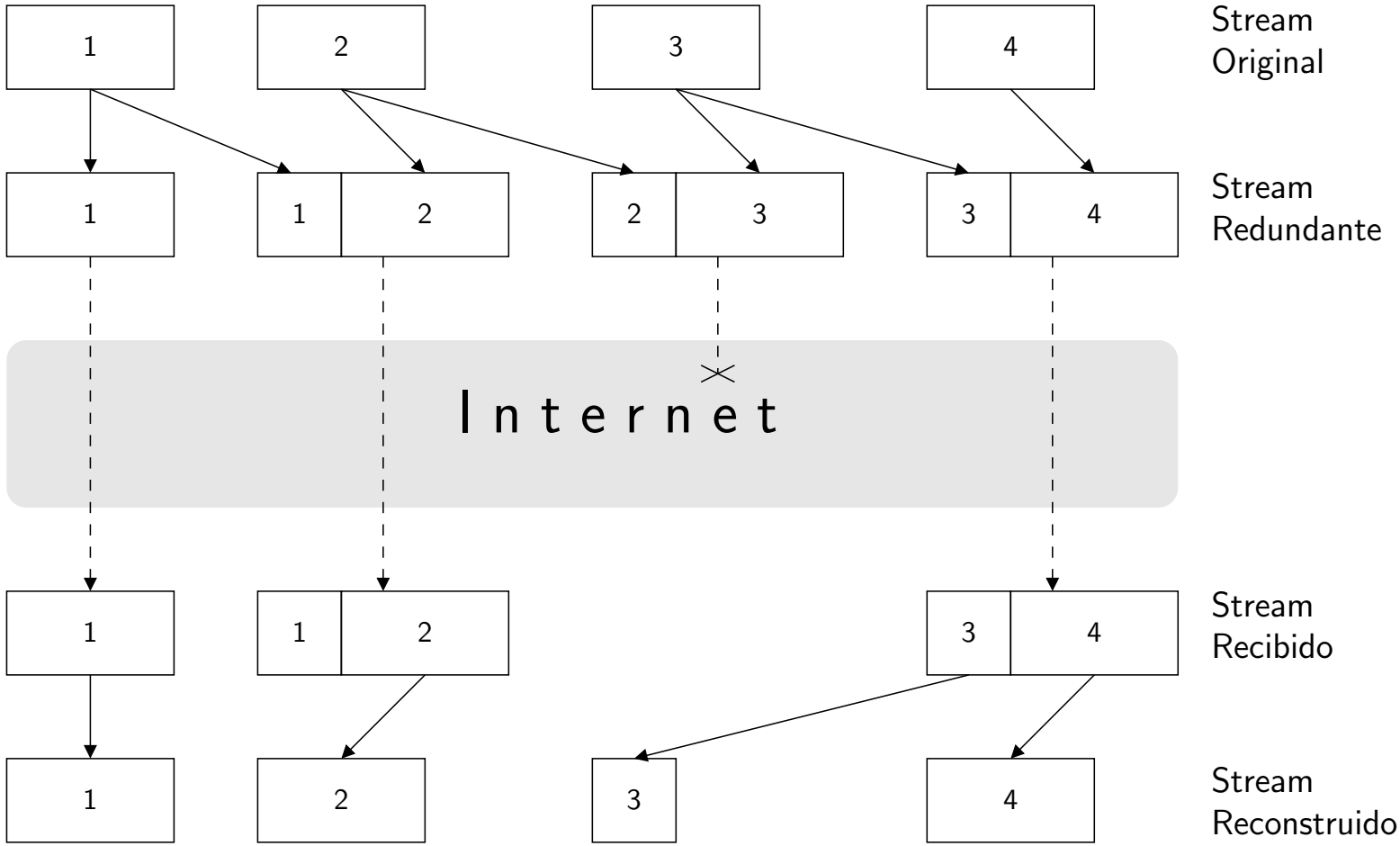
  1. Enviando información redundante. Ejemplos:

     - **Insertando un paquete de paridad periódicamente**.

       Si cada n paquetes introducimos uno con la operación XOR de los bits de mismo índice (posición) de estos paquetes, podemos reconstruir un paquete perdido tras recibir los $n+1$ paquetes.

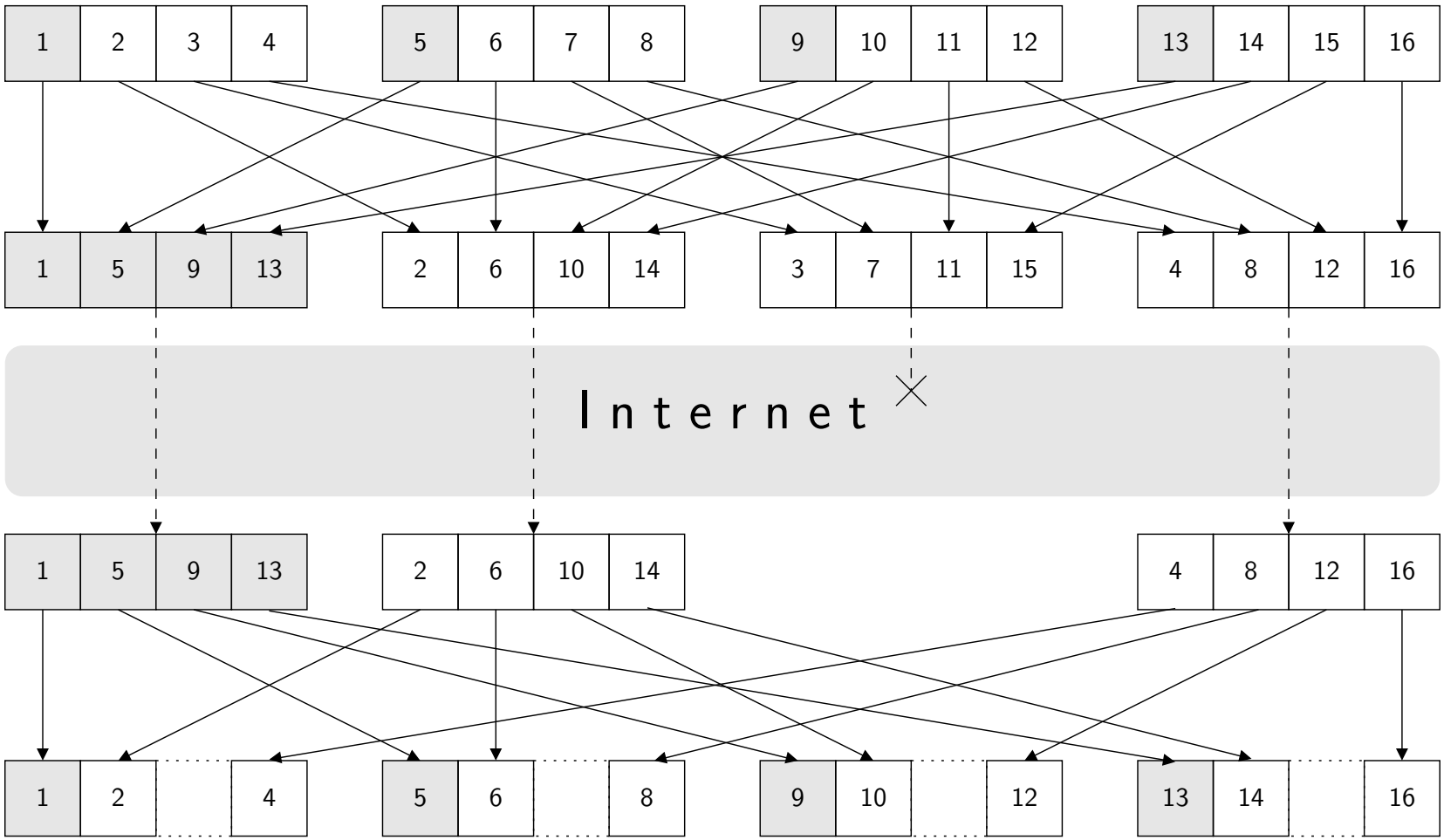
     - **Añadiendo a cada paquete información redundante sobre otro paquete (piggybacking)**.

       Podemos insertar, en cada paquete, información (de baja calidad) sobre el paquete anterior. La información piggibacked de baja calidad del paquete i se utiliza si se pierde el paquete $i-1$. Gráficamente:


1
2
3
4
Stream
Original
1
1
2
2
3
3
4
Stream
Redundante
Internet
1
1
2
3
4
Stream
Recibido
1
2
3
4
Stream
Reconstruido

2. **Entrelazando la información (interleaving) y luego aplicando alguna técnicas de predicción o interpolación.**

   El emisor coloca en cada paquete porciones de stream no consecutivos. Ejemplo:



10.5.3. Ordenación de paquetes

- ¿Por qué se desordenan los paquetes?
  - Las tablas de routing de los routers son dinámicas y por lo tanto, dos paquetes del mismo stream pueden viajar por caminos distintos con diferente longitud (en términos de tiempo).
- ¿Cómo solucionamos el problema?
  - Insertando un número de secuencia[2] en cada paquete.

[2] O una estampa de tiempo.

10.6. Protocolos para la transmisión de audio y vídeo

10.6.1. Real-Time Protocol (RTP)

- RFC 3550.
- Usa UDP.

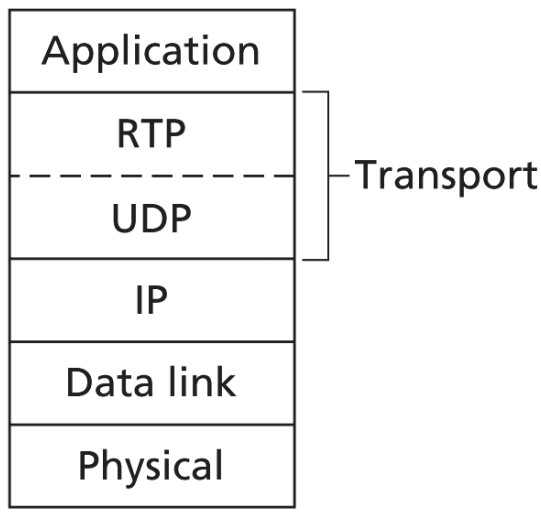
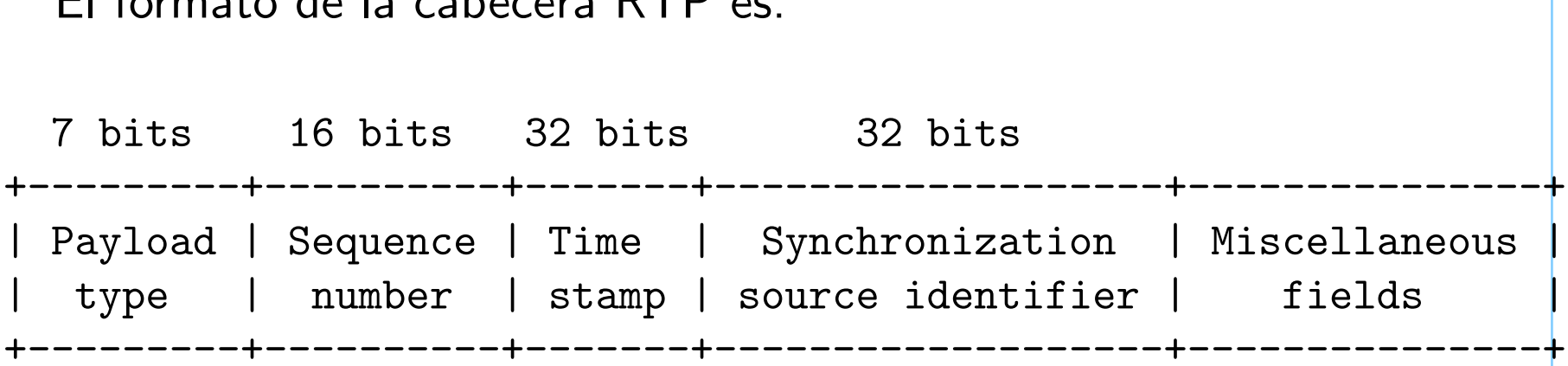


Figure 7.11 ♦ RTP can be viewed as a sublayer of the transport layer.

- Estandariza la información (como son los números de secuencia, estampas de tiempo, el algoritmo de compresión utilizado, etc.) que utilizan la mayoría de las aplicaciones de transmisión de audio y vídeo. El formato de la cabecera RTP es:



Donde:

  - `Payload type`: Indica la codificación (PCM, GSM, MPEG-1, MPEG-2, H.261, etc.). Ejemplos:

Payload-Type Number	Audio Format	Sampling Rate	Rate
0	PCM μ-law	8 kHz	64 kbps
1	1016	8 kHz	4.8 kbps
3	GSM	8 kHz	13 kbps
7	LPC	8 kHz	2.4 kbps
9	G.722	16 kHz	48–64 kbps
14	MPEG Audio	90 kHz	—
15	G.728	8 kHz	16 kbps

Table 7.1 ♦ Audio Payload Types Supported by RTP

Payload-Type Number	Video Format
26	Motion JPEG
31	H.261
32	MPEG 1 video
33	MPEG 2 video

Table 7.2 ♦ Some Video Payload Types Supported by RTP

- `Sequence Number`: Enumera los paquetes enviados.
- `Time-stamp`: indica el instante en que se generó el primer bit de datos del paquete RTP. Se utiliza para sincronizar el emisor y el receptor.
- `Synchronization source identifier`: identifica al emisor del stream (la dir IP del host no sirve porque en un host pueden generarse varios streams (audio y vídeo por separado, por ejemplo)).

10.6.2. Real-Time Control Protocol (RTCP)

- RFC 1889. Se utiliza junto con el RTP, normalmente sobre el siguiente puerto.
- Sirve para que los miembros de una sesión RTP se intercambien información (típicamente mediante multicasting).

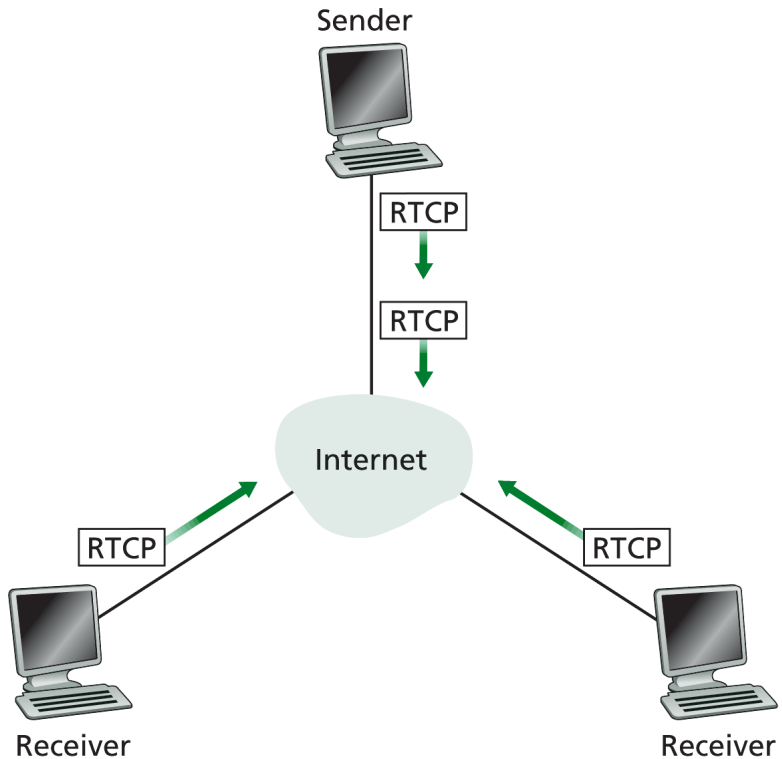


Figure 7.12 ♦ Both senders and receivers send RTCP messages.

- Periódicamente, se transmiten informes de estadísticas útiles para la transmisión de audio y vídeo: número de paquetes enviados, número de paquetes perdidos, jitter, número de hosts escuchando en una transmisiones multicast, ...
- Su uso plantea un inconveniente, sobre todo en transmisiones multicast porque el emisor puede llegar a recibir una gran cantidad de informes (uno por cada oyente). En estos casos la frecuencia de envío de los informes se decrementa en función del número de participantes en las sesiones multicast, de forma que el RTCP consume siempre aproximadamente el $5\,\%$ del ancho de banda.

10.6.3. Real-Time Streaming Protocol (RTSP)

- RFC 2326. Normalmente usa TCP.
- Se utiliza para controlar la transmisión de streams de audio y vídeo almacenados (pause, rewind, play, etc.).
- El player sabe que el stream soporta todas estas acciones porque su URL es de la forma:

```
rtsp://
```

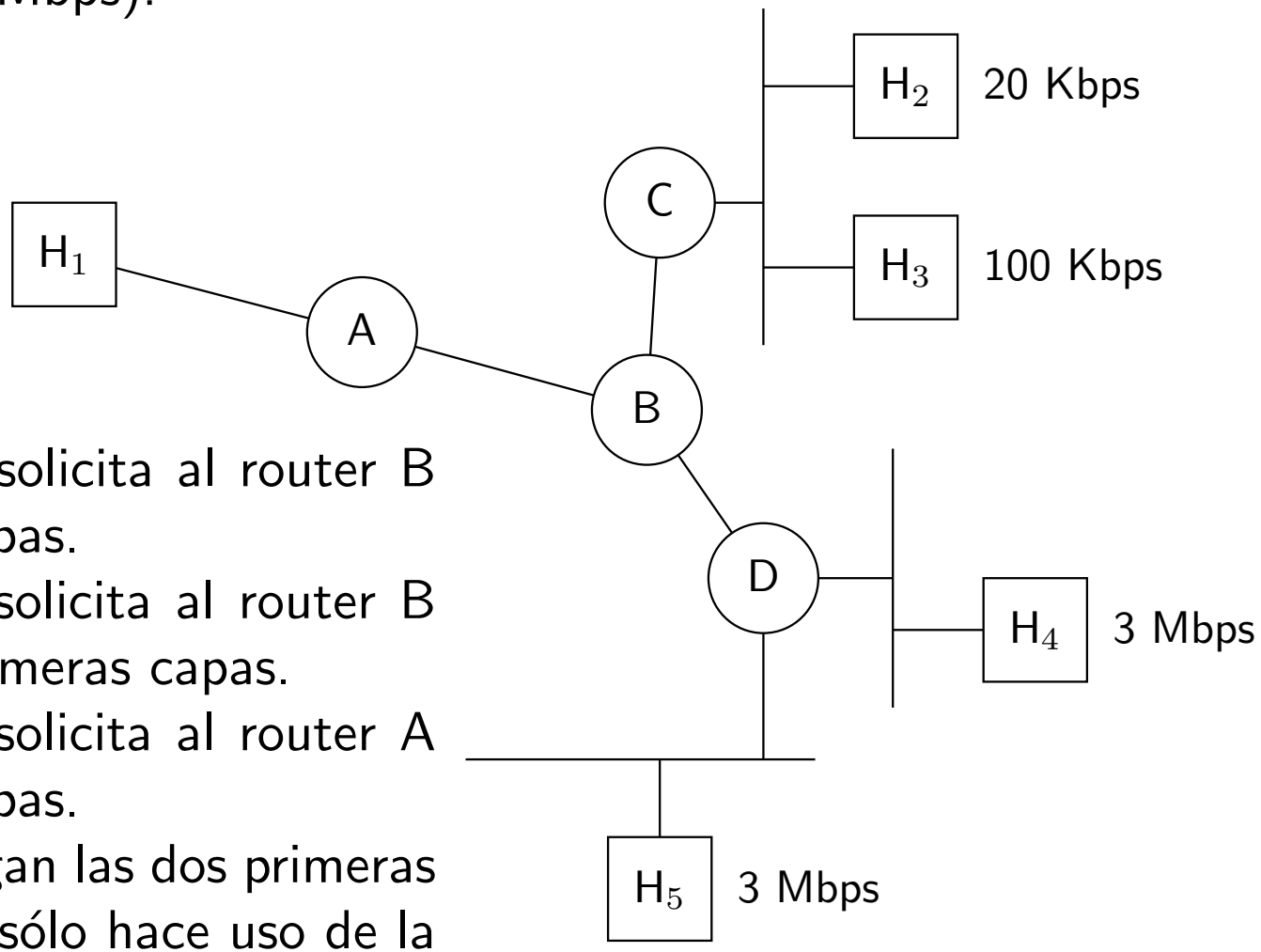
- Los mensajes RTSP son muy similares a los utilizados por el protocolo FTP o HTTP. El cliente (C:) solicita acciones y el servidor (S:) las realiza. Un ejemplo de transmisión sería:

```
C: SETUP rtsp://audio.example.com/twister/audio RTSP/1.0
   Transport: rtp/udp; compression; port=3056; mode=PLAY
S: RTSP/1.0 200 1 OK
   Session 4231
C: PLAY rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
   Session: 4231
   Range: npt=0-
S: RTSP/1.0 200 1 OK
   Session 4231
C: PAUSE rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
   Session: 4231
   Range: npt=37
S: RTSP/1.0 200 1 OK
   Session 4231
C: TEARDOWN rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
   Session: 4231
S: 200 3 OK
```

10.7. ReSerVation Protocol (RSVP)

- RFC 2205.
- Permite que los hosts realicen "reservas" de ancho de banda en Internet para transmisiones multicast.
- Las reservas expiran tras un determinado tiempo y tienen que ser periódicamente realizadas.
- El host que reserva es el que recibe, no el que envía el stream.
- Los routers tienen que entender el RSVP y permitir las reservas.
- ¿Qué es en realidad una reserva? Para el RSVP una reserva es una indicación para uno o varios routers de cúanto quiere recibir el receptor, lo que no significa que se tenga que recibir.

- El RSVP se utiliza fundamentalmente en transmisiones multicast por capas de calidad. Veamos un ejemplo donde existen 3 capas (20 Kbps, 100 Kbps y 3 Mbps):



⋆ El router D solicita al router B recibir las 3 capas.
⋆ El router C solicita al router B recibir las 2 primeras capas.
⋆ El router B solicita al router A recibir las 3 capas.
⋆ Hasta H_2 llegan las dos primeras capas, aunque sólo hace uso de la primera.

Parte III

La capa de transporte de datos

Capítulo 11

Servicios de la capa de transporte de datos

11.1. ¿Dónde corre la capa de transporte?

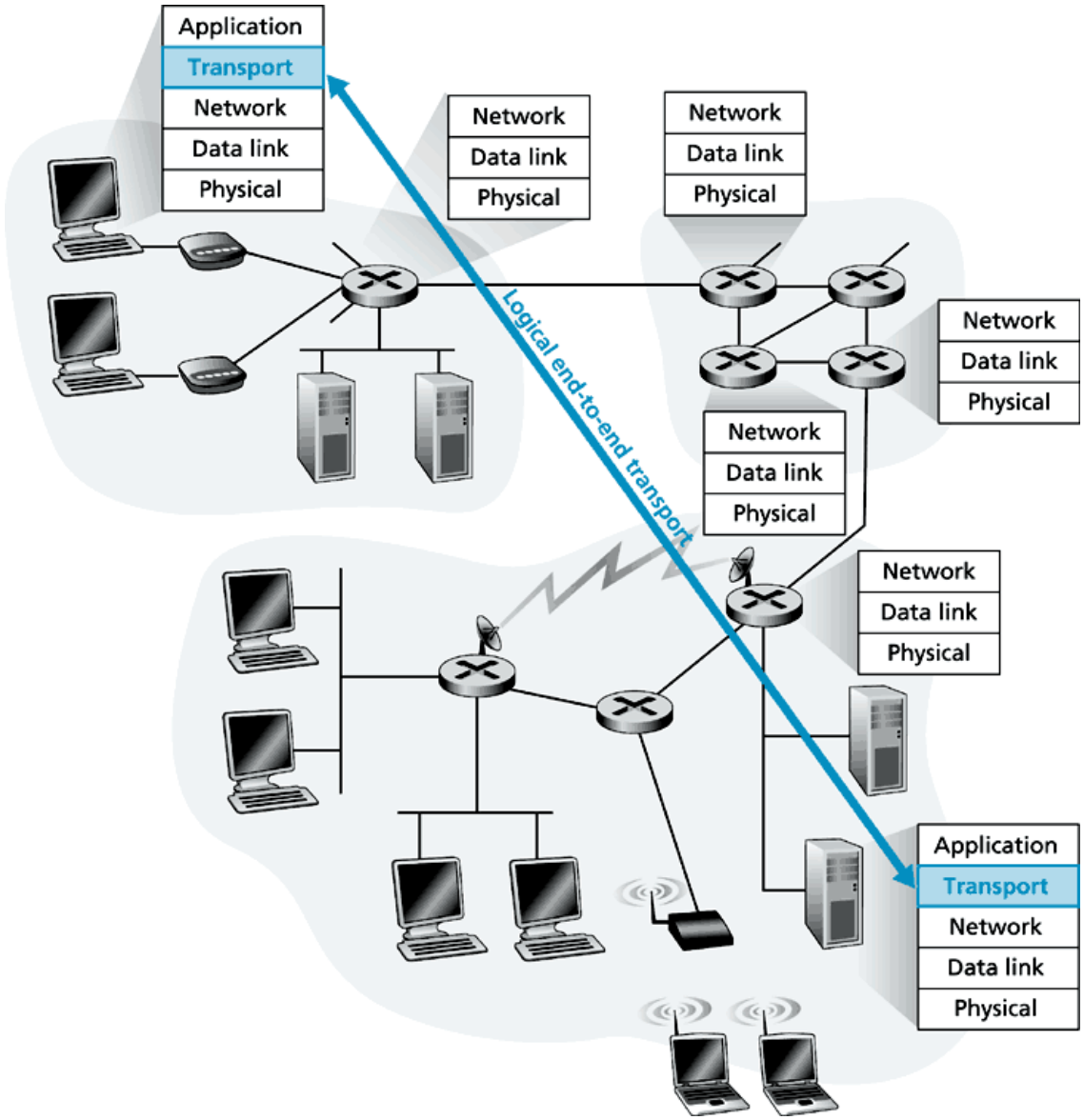
- La capa de transporte se ejecuta sólo en los sistemas finales (hosts). ¿Por qué?

  Actualmente la tasa de errores de las tecnologías de transmisión de datos son muy bajas. Esto posibilita que el control de flujo y de errores sea posible de extremo a extremo.

- Alternativamente, el control de flujo y de errores[1] se puede realizar entre cada par de nodos adyacentes. Como la tasa de error crece de forma proporcional con la distancia recorrida (número de saltos), cuando esta es suficientemente alta, es más adecuado el control salto a salto.

[1] La congestión puede entenderse en sí misma como un error de transmisión puesto que degrada la tasa de transmisión efectiva.

- Se sitúa a nivel 4, entre la capa de aplicación (nivel 5) y la capa de red (nivel 3).



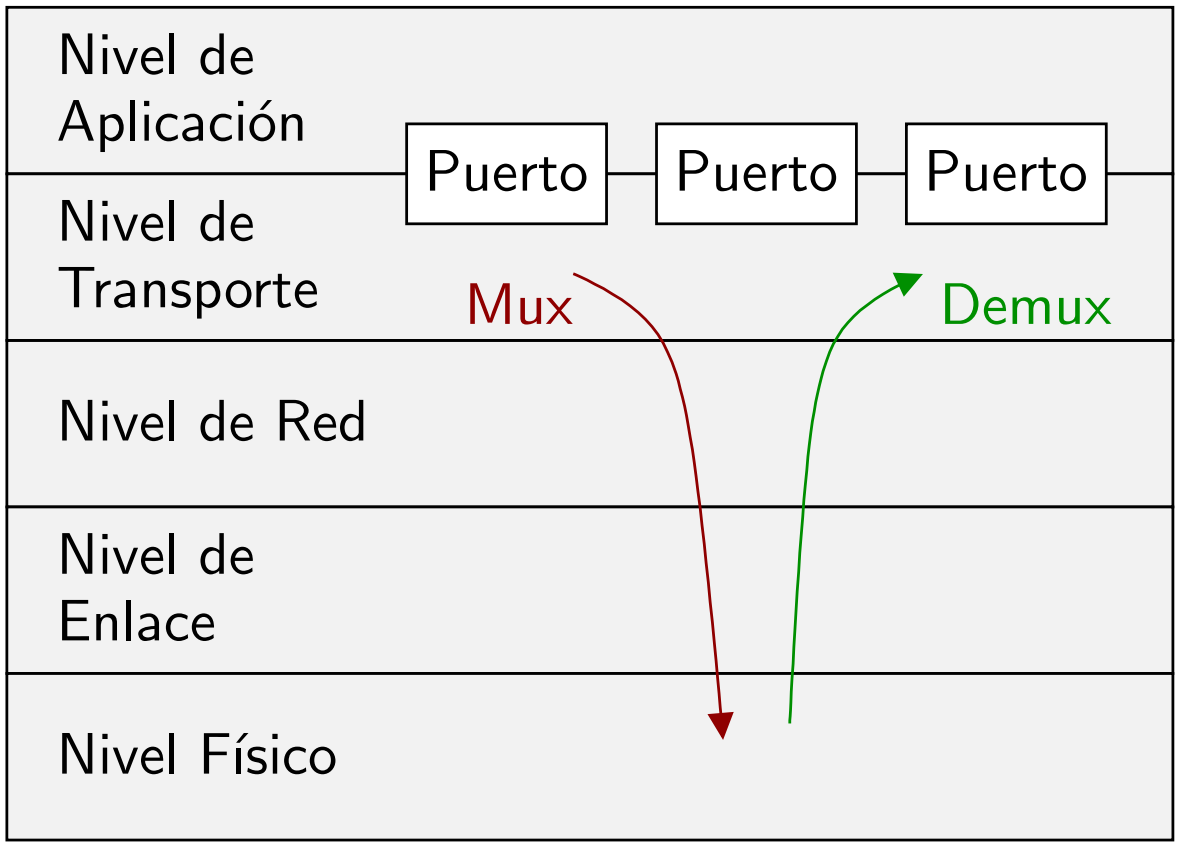
11.2. Servicios proporcionados por la capa de transporte

- Permite la **comunicación a nivel de procesos**[2] [14]. A este servicio también se le llama multiplexación. Proporcionado por el TCP y el UDP.
- Opcionalmente, corrige los errores de transmisión (modificación, pérdida, duplicación y reordenación). Sólo TCP.
- Opcionalmente, controla el flujo y la congestión. Sólo TCP.
- Opcionalmente, proporciona una conexión del tipo "byte-stream". Sólo TCP.
- No garantiza la transmisión de datos entre procesos en un tiempo limitado (principalmente porque la capa de red no garantiza la comunicación entre hosts en un tiempo limitado). TCP y UDP.

[2] Que pueden ejecutarse en una misma máquina o sobre máquinas diferentes.

11.3. El servicio de multiplexación

- En el host emisor se realiza una multiplexación (muchos procesos, una única capa de transporte/host) y en el proceso receptor una desmultiplexación (una única capa de transporte/host, muchos procesos).



- Permite aprovechar el servicio de entrega de host-a-host proporcionado por la capa de red para ofrecer un servicio de entrega de proceso-a-proceso.
- Los procesos que se ejecutan dentro de un host son diferenciados por la capa de transporte a partir del puerto en el que están escuchando.
- En los paquetes de datos figurará el puerto destino.

11.4. Sobre los puertos

- Permiten que más de una aplicación utilice la red en un determinado intervalo de tiempo.
- Cada puerto tiene asignado un número entre 0 y 65.535.
- Los puertos que comprenden desde el 0 al 1.023 están reservados para aplicaciones estándares (como la Web que utiliza el puerto 80) [4]. Dicha lista es mantenida por el IANA (Internet Assigned Numbers Authority, `http://www.iana.org`) y se puede consultar en el RFC 1700.[3]

[3] En Unix, esta información el S.O. la almacena en el fichero `/etc/services`.

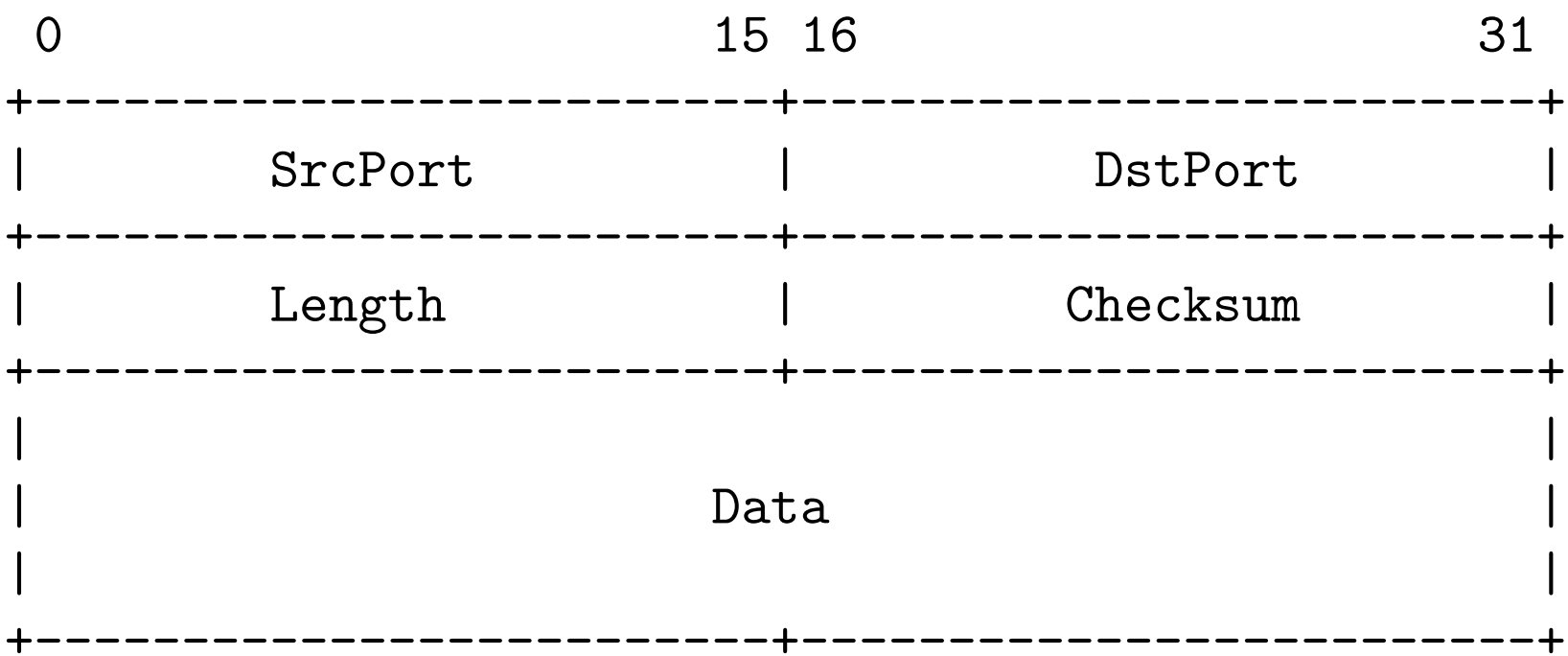
Capítulo 12

El UDP (User Datagram Protocol)

12.1. El UDP

- RFC 768.
- Multiplexa [14] y desmultiplexa [4].
- Transmite datagramas (paquetes de datos que son transmitidos independientemente y sin previo establecimiento de conexión).
- Opcionalmente, comprueba si se han producido errores de transmisión (no los corrige).
- Cuando se emplea el UDP sólo un proceso (receptor o emisor) puede utilizar cada puerto.

12.2. Formato del datagrama UDP



`SrcPort` = puerto del proceso emisor.
`DstPort` = puerto del proceso receptor.
`Length` = longitud en bytes del datagrama UDP (cabecera y datos).
`Checksum` = checksum de la cabecera, pseudo-cabecera y datos.
`Data` = datos a trasmitir (hasta 64 KB).

El checksum es opcional.

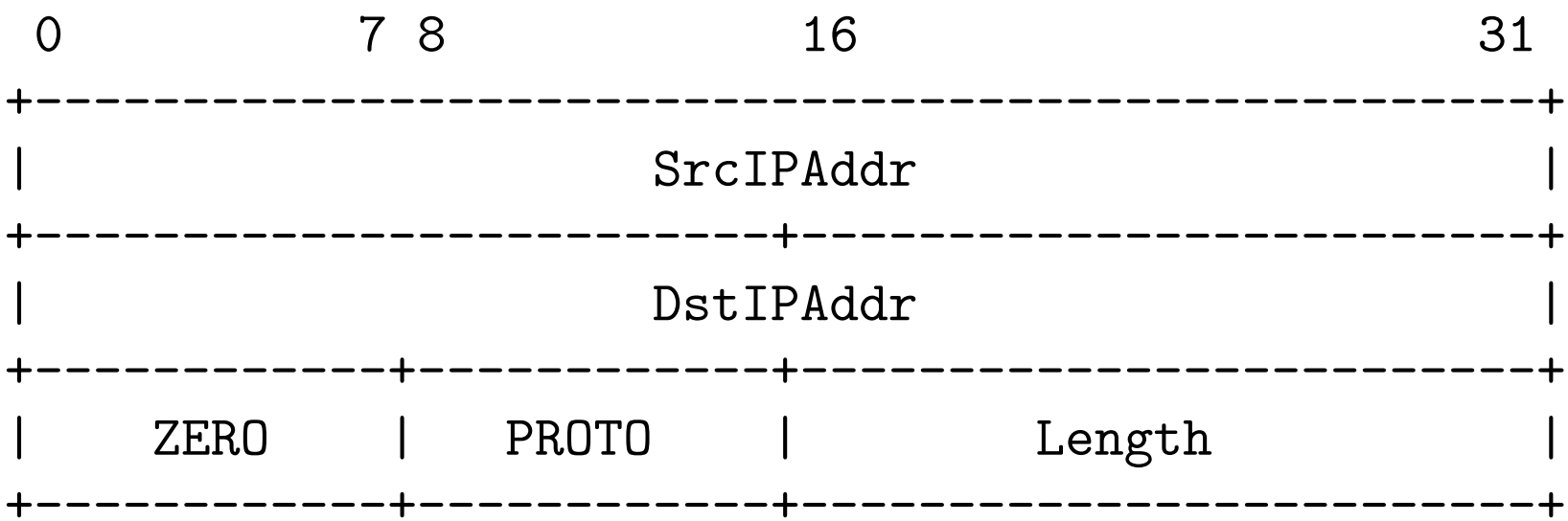
1. Si `checksum` $= 0$ significa que no se calcula.
2. Si `checksum` $\neq 0$, se calcula teniendo en cuenta la pseudo-cabecera, la cabecera del datagrama UDP y los datos a transmitir.

El puerto del proceso emisor se utiliza para que el proceso receptor pueda contestar.

12.3. La suma de comprobación (checksum)

- RFC 1071.
- Forma simple de control de errores.
- Consiste en calcular la suma aritmética módulo 16 de todas las palabras del datagrama UDP y de la pseudo-cabecera, y calcular su complemento a 1 (negativo).
- Se puede ignorar, permitiendo así la recepción de un datagrama con errores [14].

- La pseudo-cabecera no se transmite. Se utiliza en el checksum para verificar, entre otras cosas, que el datagrama UDP llega al destino correcto y que el host origen es el que indica el IP. Su contenido es:



`SrcIPAddr` = dirección IP del host emisor (proporcionada por el IP).
`DstIPAddr` = dir. IP del host receptor (prop. por la aplicación).
`ZERO` = byte de relleno igual a 0.
`PROTO` = identificador de protocolo (17 para UDP).
`Length` = tamaño del datagrama UDP.

- El receptor realiza dicha suma, pero usando además la suma de comprobación. Si el resultado es 0, entonces presumiblemente no existen errores de transmisión.

12.4. ¿Cuándo usar el UDP?

Cuando queremos minimizar la latencia

- Cuando usamos el UDP no existe establecimiento de conexión. En aplicaciones como el DNS esto es fundamental.
- Con el UDP el emisor controla exactamente cuándo los datos son enviados.

Cuando se permite la pérdida de datos

- Existen aplicaciones donde la no llegada de datos es peor que la llegada de ellos aunque modificados (probablemente en unos pocos bits). En aplicaciones como en la transmisión de audio y vídeo esto es importante.

Cuando queremos hacer transmisiones multicast

- UDP es ideal para transmitir de uno a muchos porque la replicación del datagrama UDP por parte de los routers no implica ninguna modificación en el comportamiento del emisor (en concreto, la tarea de emitir en unicast o en multicast es básicamente la misma e independiente del número de receptores a los que llega el paquete). En aplicaciones como la difusión de audio y vídeo esto es esencial.

Cuando queremos controlar el flujo

- Usando UDP, la aplicación sabe qué datos son enviados en cada instante. En aplicaciones como en la transmisión de flujos de audio el control de flujo lo impone la aplicación, no la capa de transporte.

Application	Application-Layer Protocol	Underlying Transport Protocol
Electronic mail	SMTP	TCP
Remote terminal access	Telnet	TCP
Web	HTTP	TCP
File transfer	FTP	TCP
Remote file server	NFS	Typically UDP
Streaming multimedia	typically proprietary	Typically UDP
Internet telephony	typically proprietary	Typically UDP
Network management	SNMP	Typically UDP
Routing protocol	RIP	Typically UDP
Name translation	DNS	Typically UDP

Figure 3.6 ♦ Popular Internet applications and their underlying transport protocols

12.5. Sobre el control de la congestión

- El UDP no realiza control de la congestión.
- Transmitir datos de forma masiva usando UDP es una forma potencialmente peligrosa de congestionar la red.

Capítulo 13

Transferencia fiable de datos

13.1. La transferencia fiable de datos

- Se utiliza para transmitir datos sin errores de transmisión y para controlar el flujo[1].
- Se implementa en diferentes niveles [14], dependiendo de las necesidades:
  1. **Capa de aplicación.** Típicamente porque ningún nivel inferior la implementa o no lo hace adecuadamente.
  2. **Capa de transporte.** Es el lugar natural de implementación (TCP) porque una gran cantidad de aplicaciones necesitan garantizar la entrega correcta.
  3. **Capa de enlace de datos.** Se implementa a este nivel cuando la tasa de errores de un enlace es muy alta y ha de realizarse el control, nodo a nodo.

[1] otra forma de evitar errores.

13.2. Detección y corrección de errores

- Los códigos de detección de errores introducen redundancia y los de corrección de errores más aún.
- Los códigos de detección de errores no indican los errores, sólo que se han producido.
- Los códigos de corrección de errores indican los errores y por tanto se pueden corregir.
- Ninguno de estos códigos resuelve los problema de la pérdida de paquetes de datos y/o su desordenación.

13.3. Protocolos ARQ

- Los protocolos de solicitud de repetición automática o protocolos ARQ (Automatic Repeat reQuest) retransmiten aquellos paquetes de datos que han llegado con errores.
- Utilizan códigos de detección de errores.
- Nacieron como una alternativa automatizada a la "verificación de eco"[2] [10].

[2] Método de control manual de errores que se utiliza en programas de acceso remoto como Telnet. Consiste en una técnica muy sencilla: cuando el usuario teclea un carácter, éste es transmitido hasta la computadora y se reenvía de nuevo hacia el terminal que lo presenta en pantalla. Si el usuario se da cuenta de que lo que se muestra no coincide con lo que él esperaba, es el usuario el encargado de realizar la corrección de errores oportuna usando los caracteres de control de los que dispone el terminal.

13.4. El protocolo ARQ con parada y espera (stop and wait)

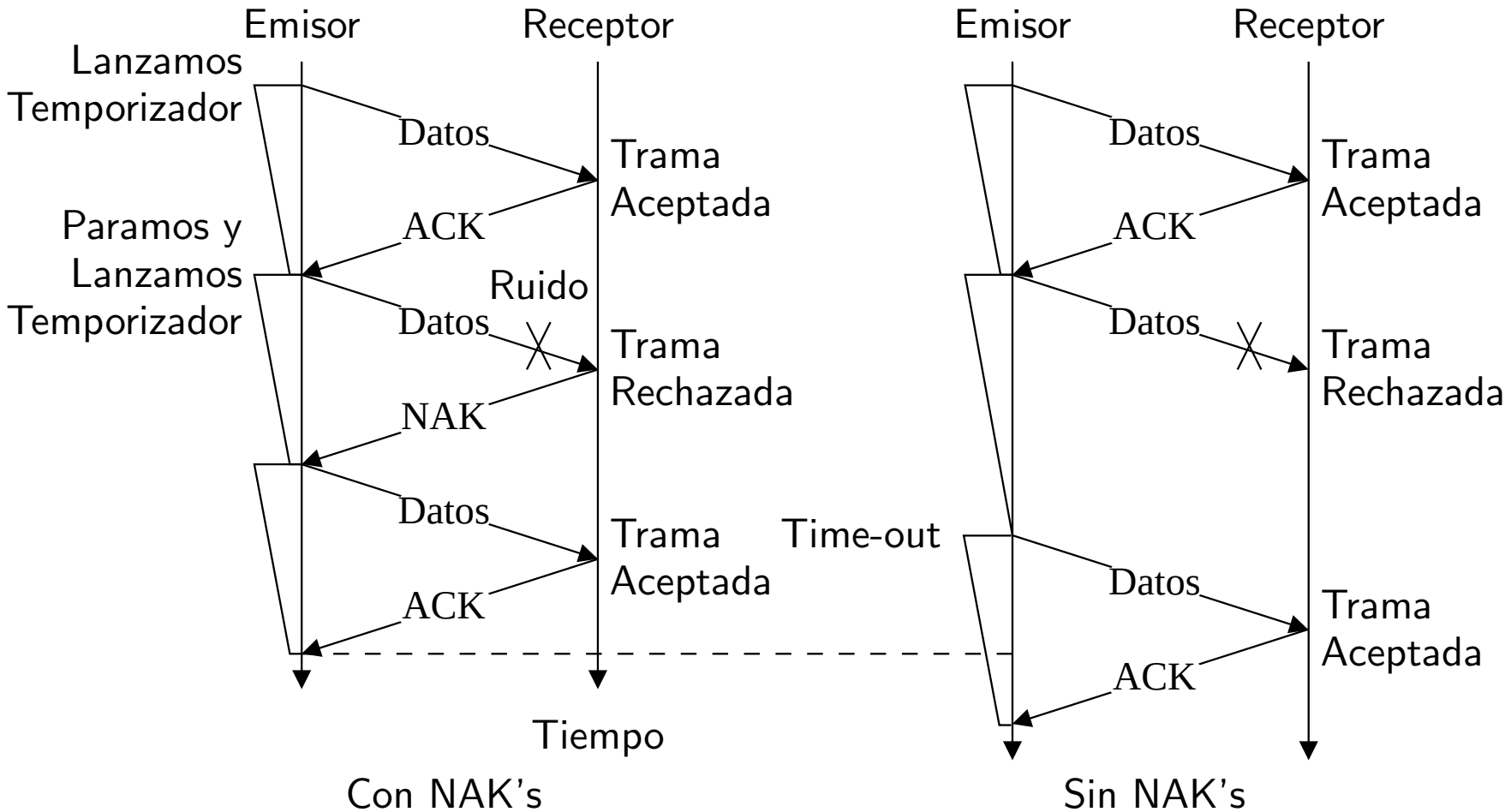
- Se trata del protocolo ARQ más sencillo. Se basa en las siguientes reglas:

  1. El emisor envía un paquete y espera a que el receptor confirme positiva o negativamente su recepción.
  2. El receptor envía un reconocimiento positivo[3] o ACK (ACKnowledgement) cuando el paquete ha llegado sin errores.
  3. El receptor envía un reconocimiento negativo o NAK (Negative AcKnowledgement) cuando el paquete ha llegado con errores, aunque esto es opcional[4].
  4. El emisor lanza un temporizador para cada paquete enviado con el objetivo de saber cuánto tiempo debe esperar una confirmación.

[3] Algo similar a un acuse de recibo en el correo ordinario.

[4] Sin embargo, su uso mejora el rendimiento [8].

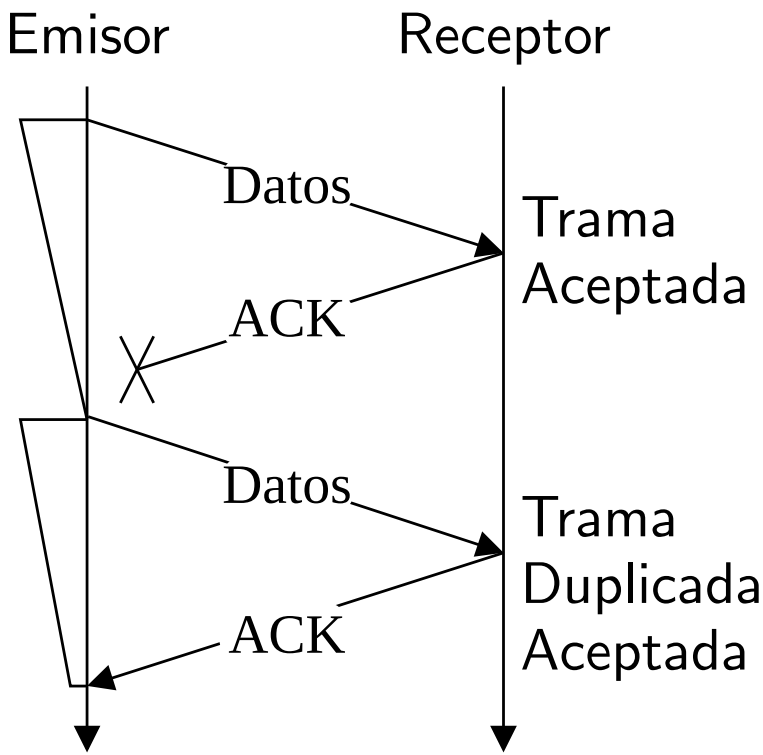
13.4.1. NAK vs sólo-ACK

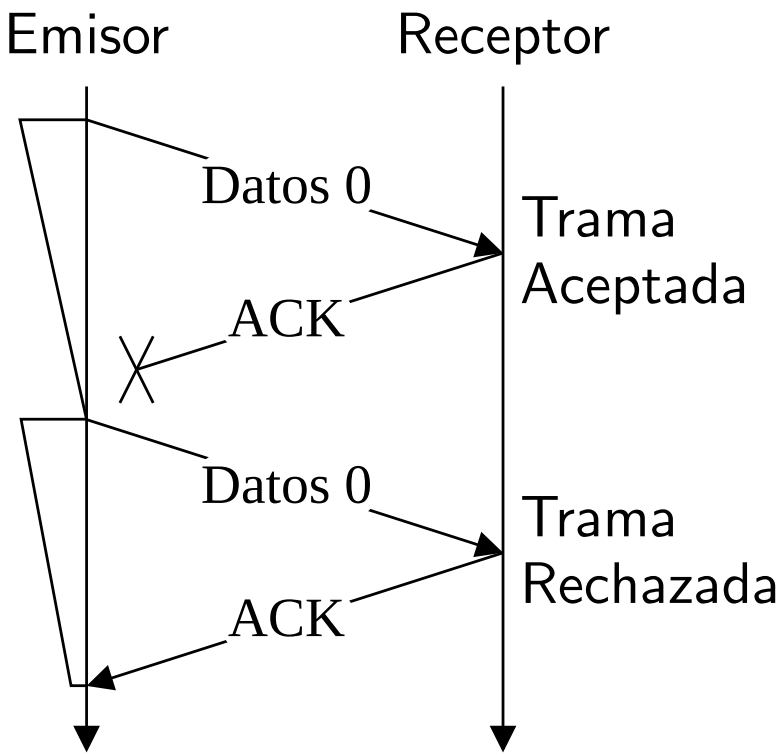


¡Nótese que la versión que utiliza NAK's es más rápida!

13.4.2. Numeración de los paquetes

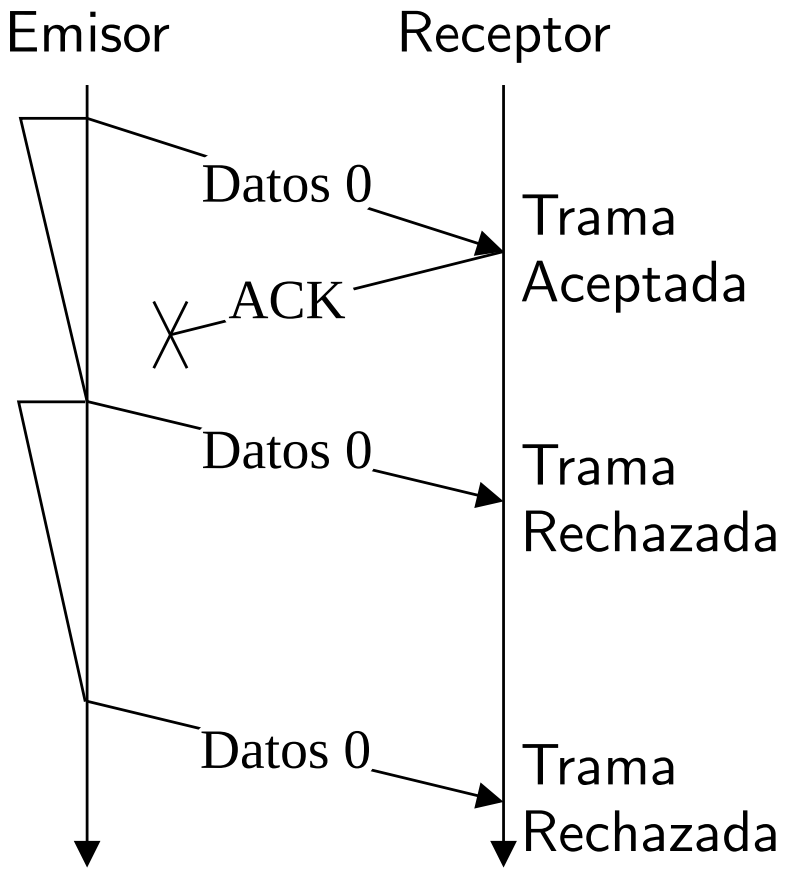
- Todos los paquetes deben ser enumerados para evitar duplicados.

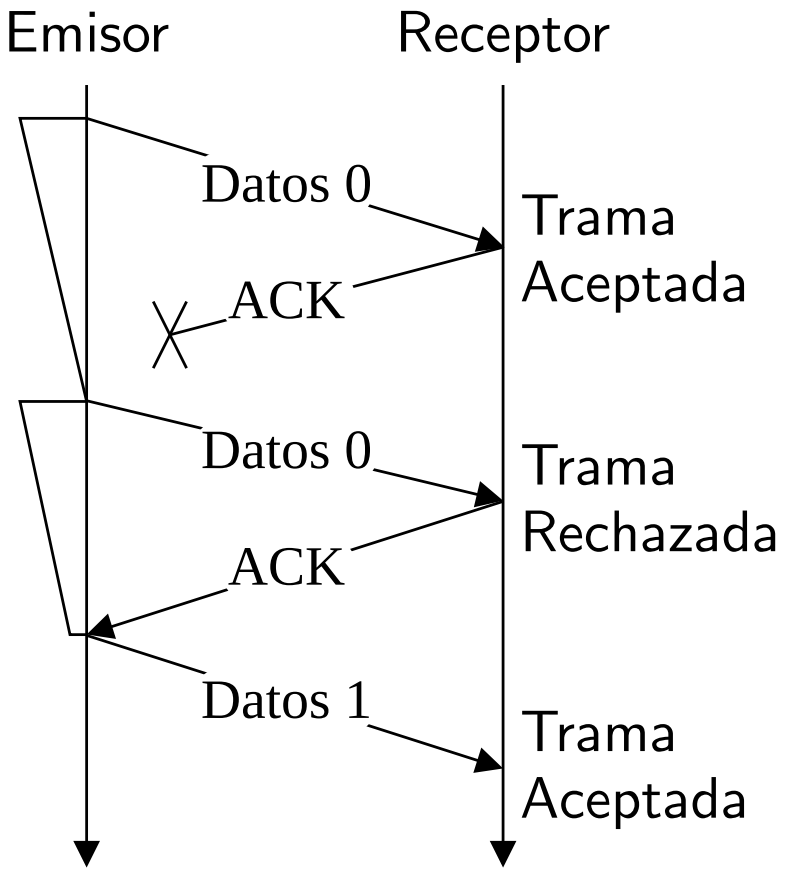




13.4.3. Confirmación de los paquetes duplicados

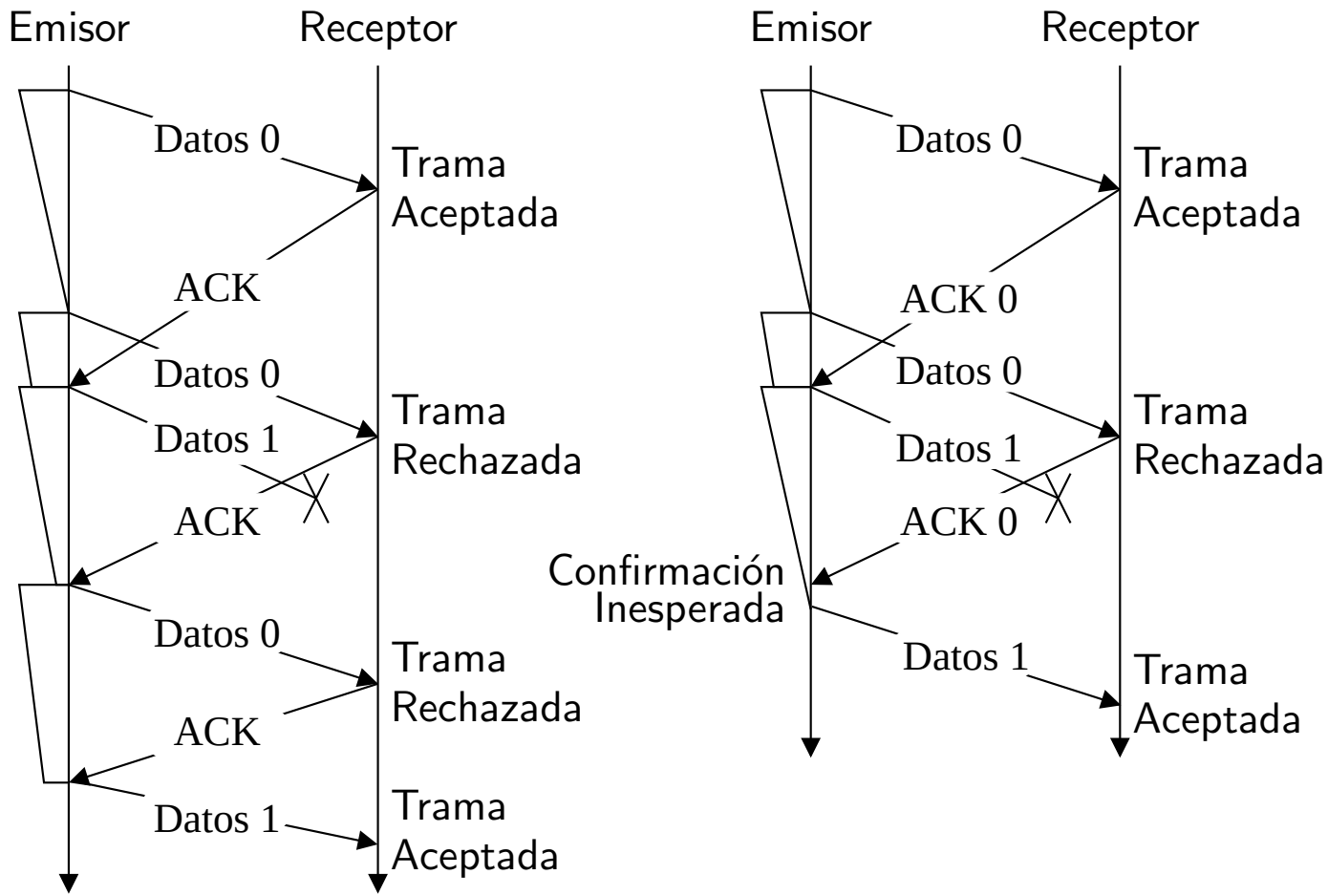
- Los paquetes duplicados deben confirmarse positivamente para evitar interbloqueos.





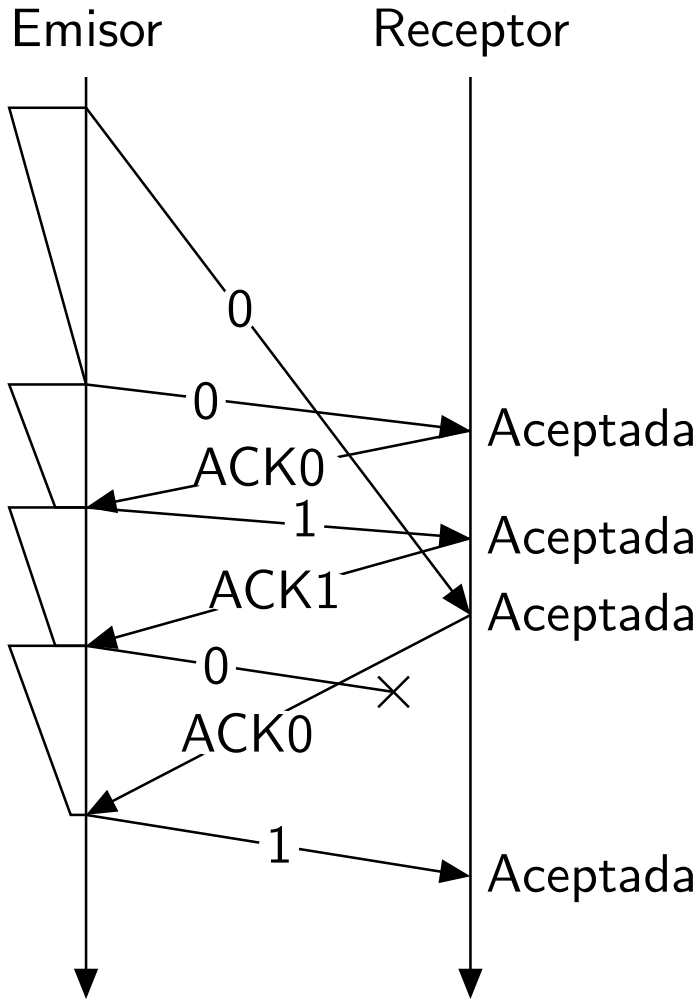
13.4.4. Numeración de los paquetes de confirmación

- Los paquetes de confirmación deben enumerarse también para evitar la pérdida de datos. En el ejemplo (izq), el primer "Datos 1" se pierde.



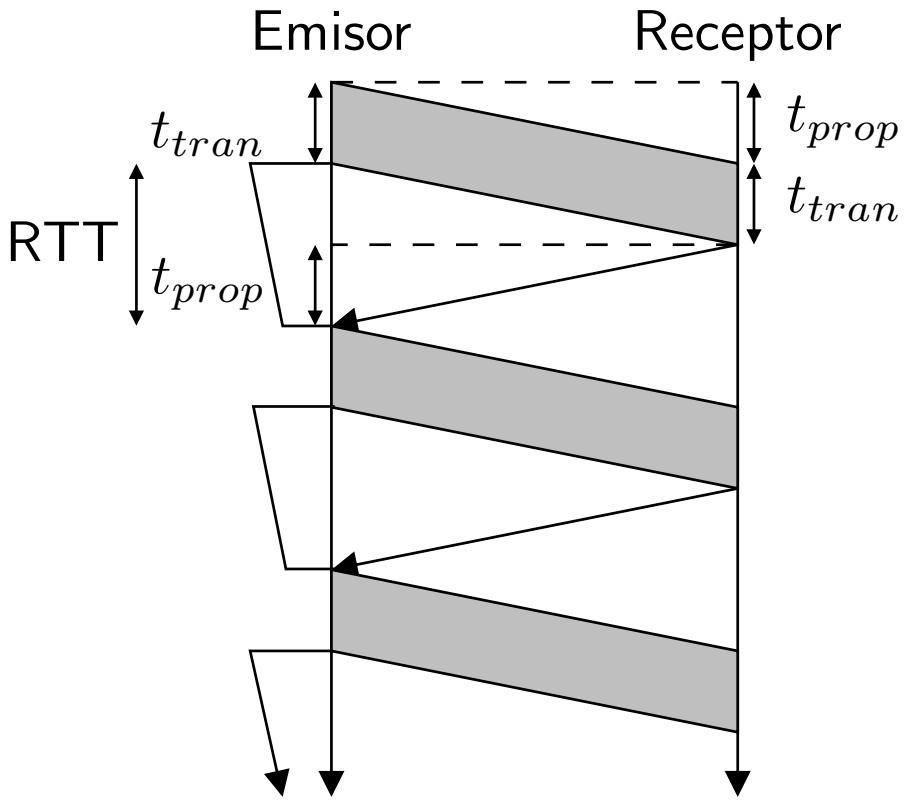
13.4.5. El protocolo falla si los paquetes se desordenan

- El enlace de transmisión debe ser punto a punto, es decir, los paquetes no pueden desordenarse [32].

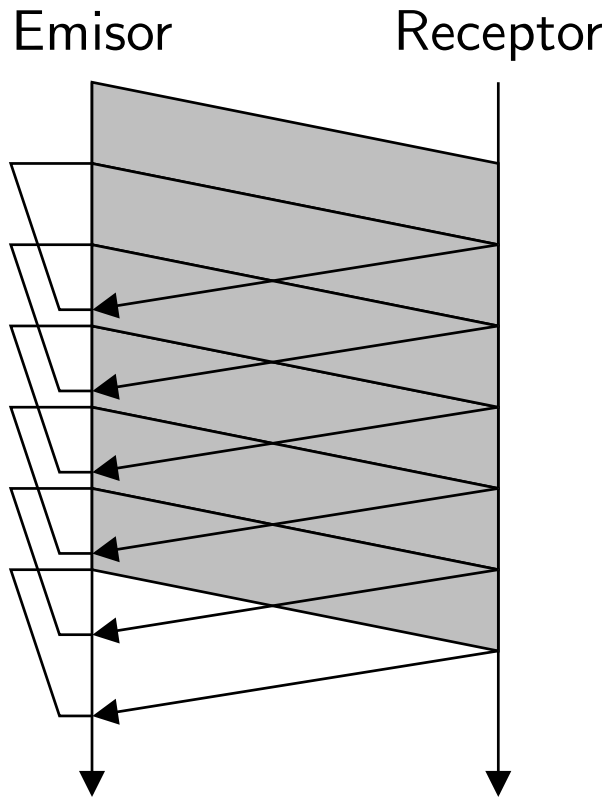


13.4.6. Rendimiento pobre

- ARQ con parada y espera no es adecuado cuando los retrasos de extremo a extremo y las tasas de transmisión de los enlaces son altas [14, 22].

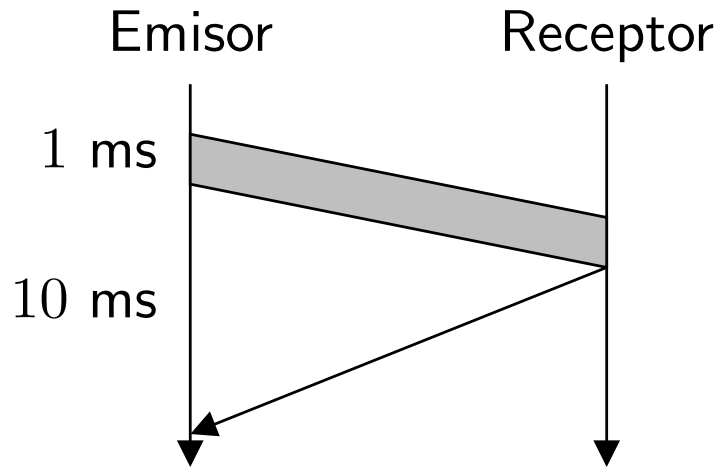


Parada y Espera



Protocolo Ideal

Ejemplo 13.11: Supongamos que se están transmitiendo paquetes de 1.000 bits a través de un canal de 1 Mbps y que el RTT del enlace es de 10 ms. En la figura



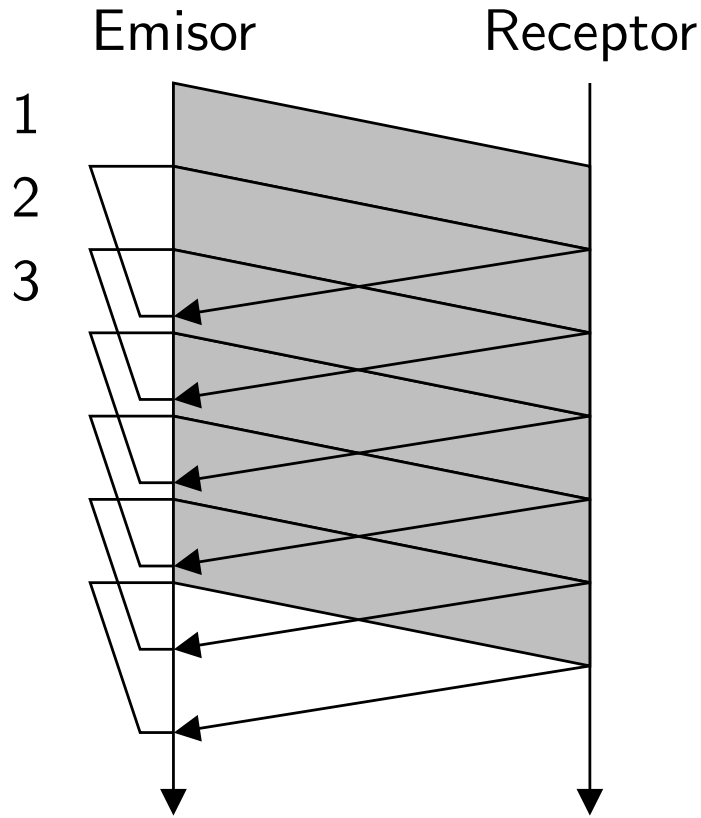
se muestra el time-line asociado a la transmisión de un paquete de datos. El emisor necesita

$$t_{tran} = \frac{10^3 \text{ b/paquete}}{10^6 \text{ b/s}} = 1 \text{ ms/paquete.}$$

Como cada 11 ms se logran enviar 1.000 bits, es lógico pensar que cada 11 s se lograrían enviar 10^6 bits. Por lo tanto la transmisión es 11 veces más lenta de lo debería ser.

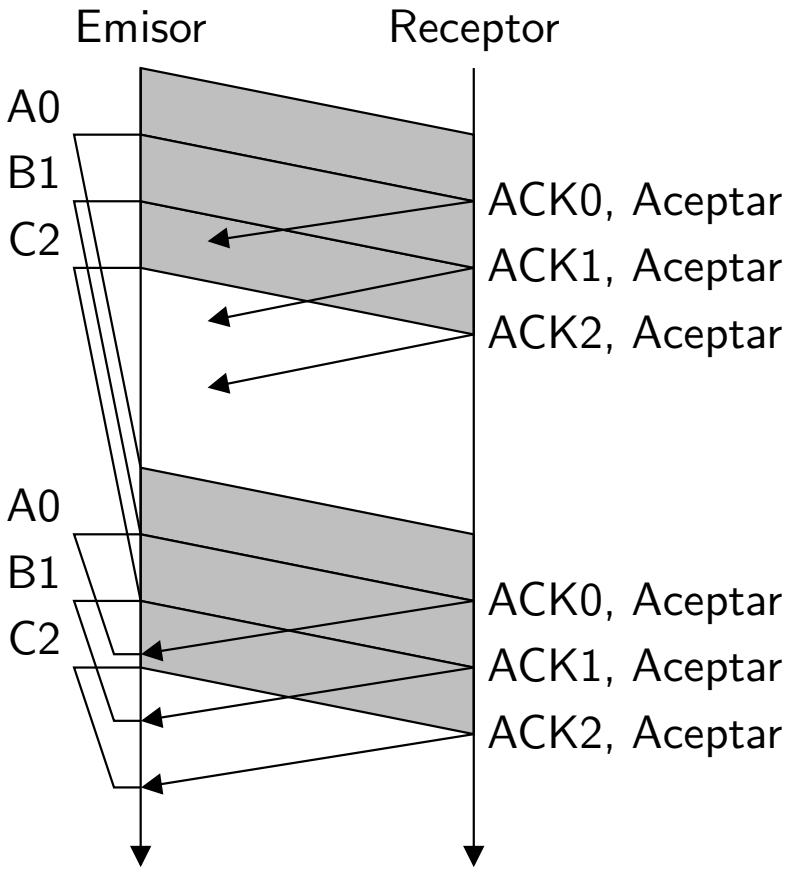
13.5. ARQ con retroceso a n (go back n)

- En GBN, el emisor puede transmitir tantos paquetes (n) como sean necesarios para que no se tenga que esperar (dejando de transmitir datos) a un reconocimiento. A esto se le llama pipelining (véase el applet http://www.ace.ual.es/~vruiz/docencia/redes/teoria/applets/gbn.html).
- Ejemplo ($n = 3$):

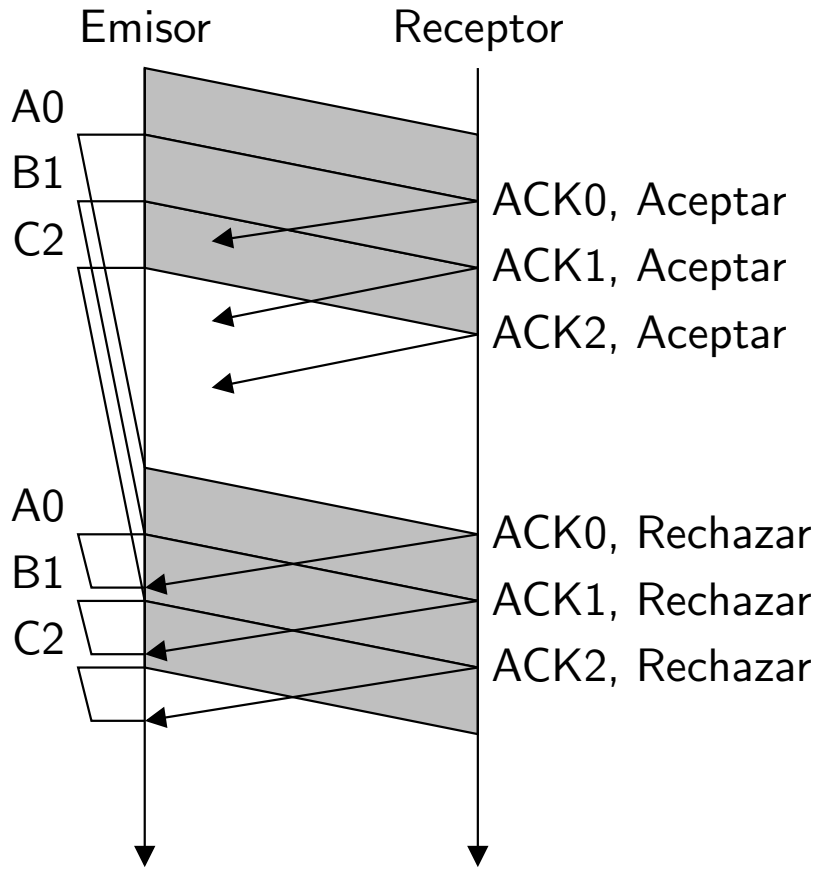


13.5.1. Longitud de la secuencia de conteo

- La longitud l de la secuencia de conteo debe ser mayor que el número de paquetes n sin confirmar ($l > n$):



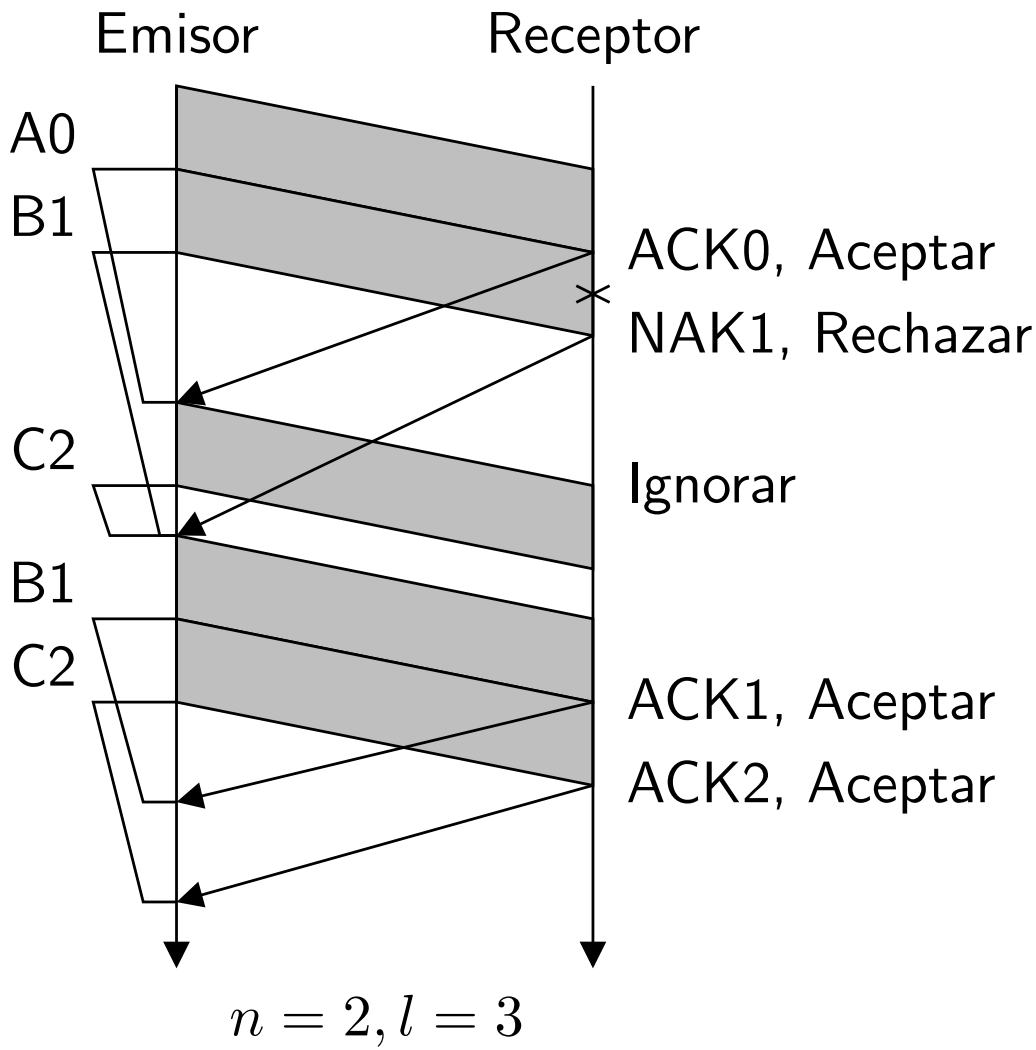
$n = l = 3$



$n = 3, l = 4$

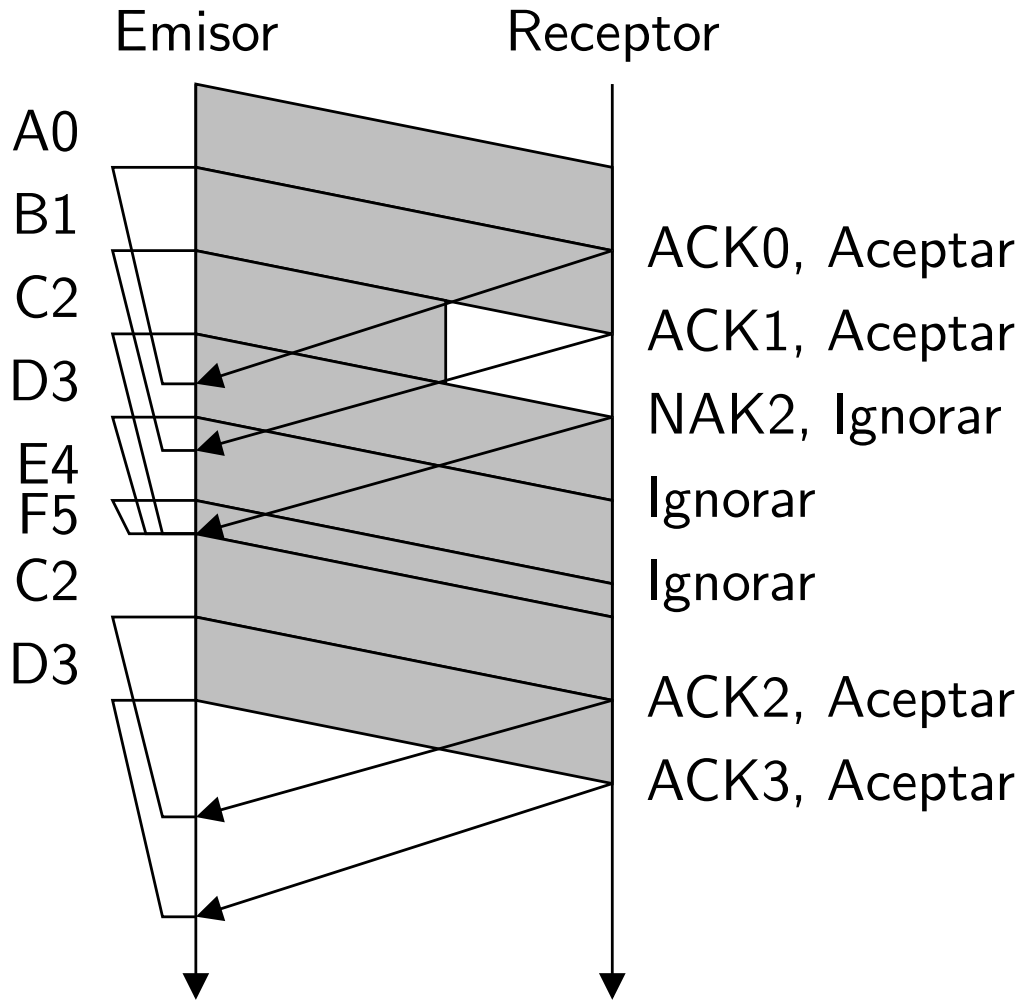
13.5.2. Tratamiento de los errores

- Un NAKk significa que hay que retransmitir el paquete k y siguientes.
- Ejemplo:



$n = 2, l = 3$

- Ejemplo:



$n = 5, l = 6$

- Los paquetes de datos que llegan fuera de secuencia, es decir, que no están precedidos de los paquetes anteriores, son descartados. Esto se hace porque de todas formas el emisor los va a retransmitir.

13.5.3. Confirmación acumulativa

- Minimiza el número de confirmaciones:

  1. Un ACKk significa que todos los paquetes enviados antes que el k-ésimo paquete han llegado con éxito.
  2. Un NAKk significa que todos los paquetes enviados antes que el k-ésimo han llegado con éxito, pero el paquete k debe ser retransmitido.

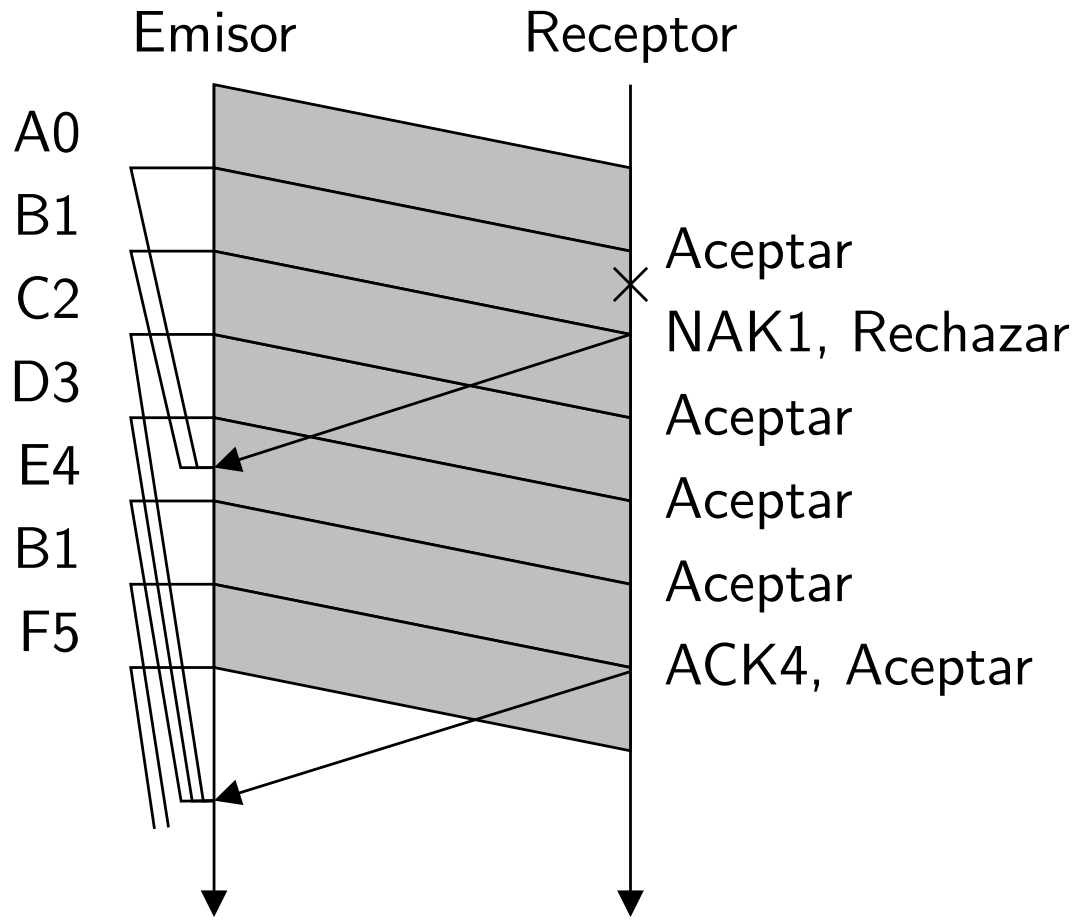
13.5.4. Piggybacking [32]

- En la práctica muchas veces tanto el emisor como el receptor son a su vez receptor y emisor (respectivamente). Una forma de disminuir el consumo de ancho de banda en este caso es transmitir las confirmaciones dentro de los paquetes de datos si estos se envían con la suficiente frecuencia.

13.6. ARQ con repetición selectiva (selective repeat o SR)

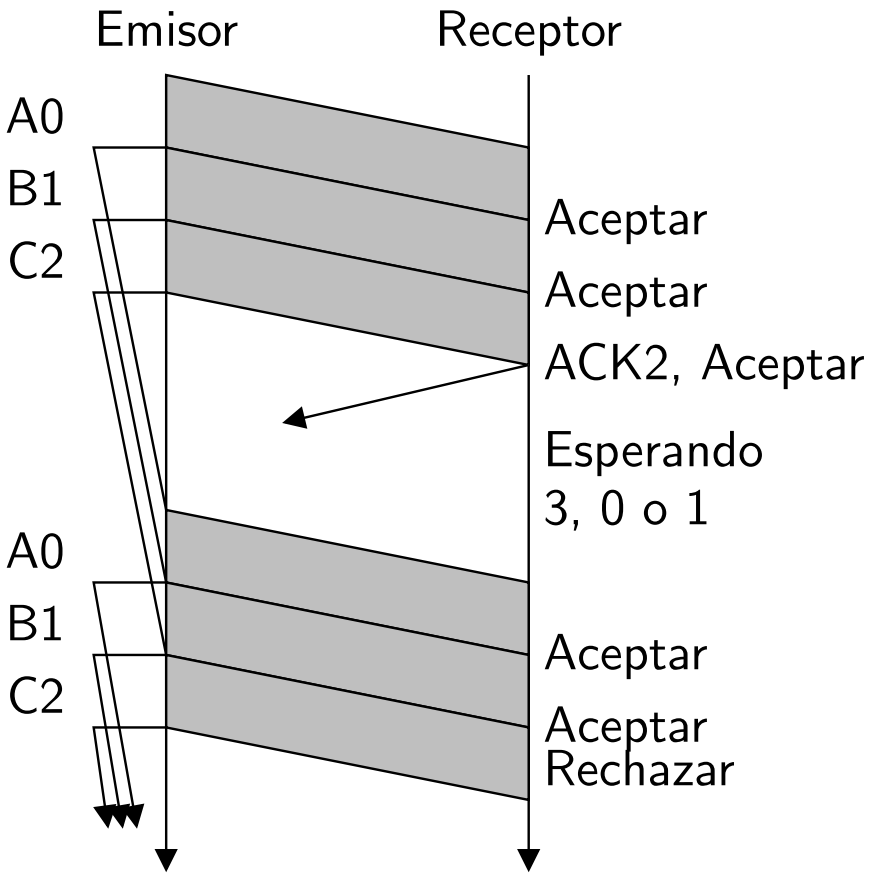
- GBN presenta un rendimiento pobre cuando los enlaces tienen altas tasas de transmisión, altas latencias y relativamente altas tasas de errores porque un error en un único paquete de los que están viajando (que pueden ser muchos) implica la retransmisión de todos los que lo suceden en el enlace (que ya han sido transmitidos) [32].

- SR retransmite sólo aquellos paquetes que han llegado con errores, es decir, cuando se recibe un NAKk el emisor retransmite sólo el paquete k. Ejemplo:

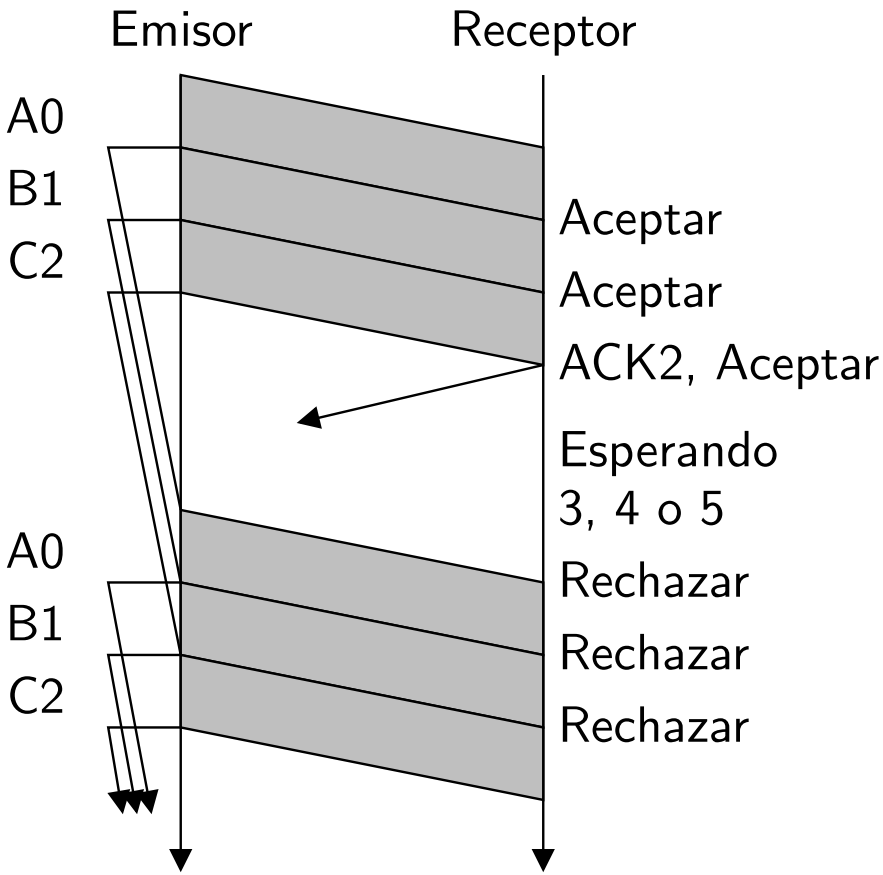


13.6.1. Longitud de la secuencia de conteo

- La longitud de la secuencia de conteo l debe ser al menos el doble que el número máximo n de paquetes enviados sin confirmación ($l \geq 2n$). En el ejemplo (izq.) A0 y B1 se duplican:



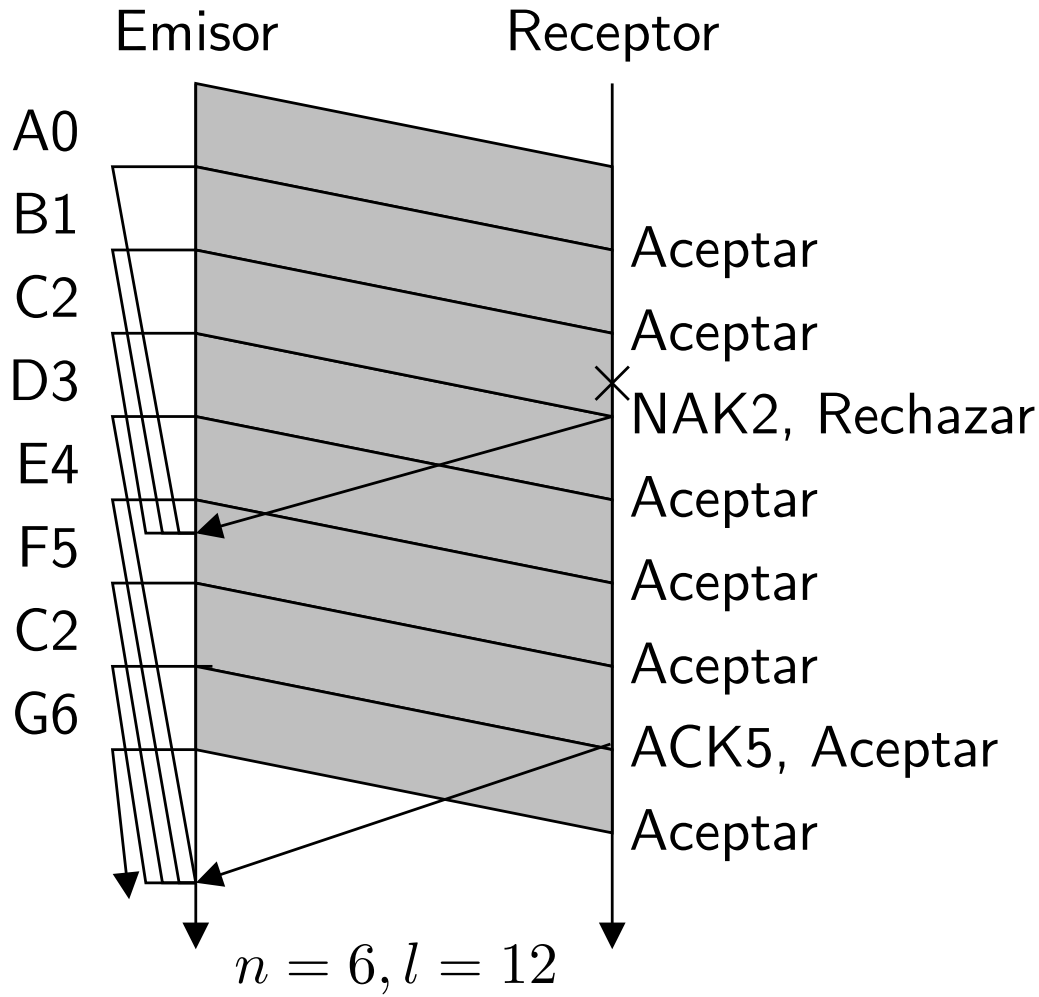
$n = 3, l = 4$

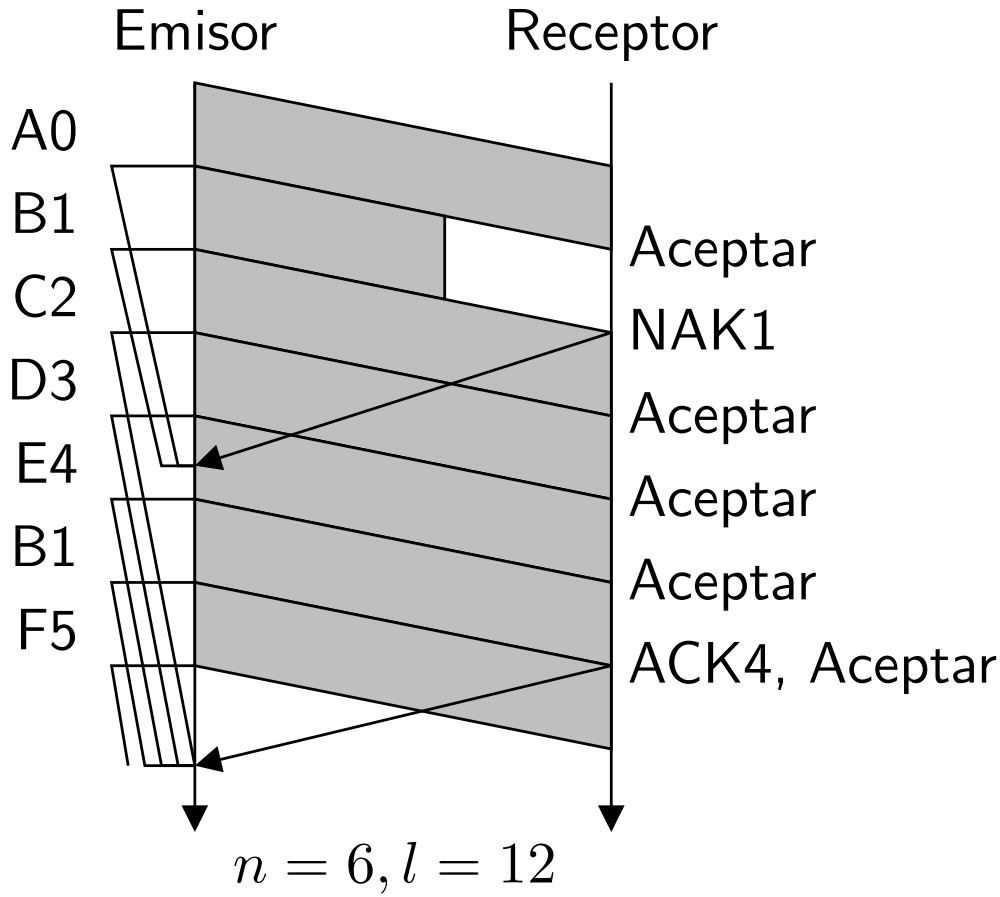


$n = 3, l = 6$

Tratamiento de los errores (otros ejemplos)

- Un NAKk significa que hay que retrasmitir sólo el paquete k:

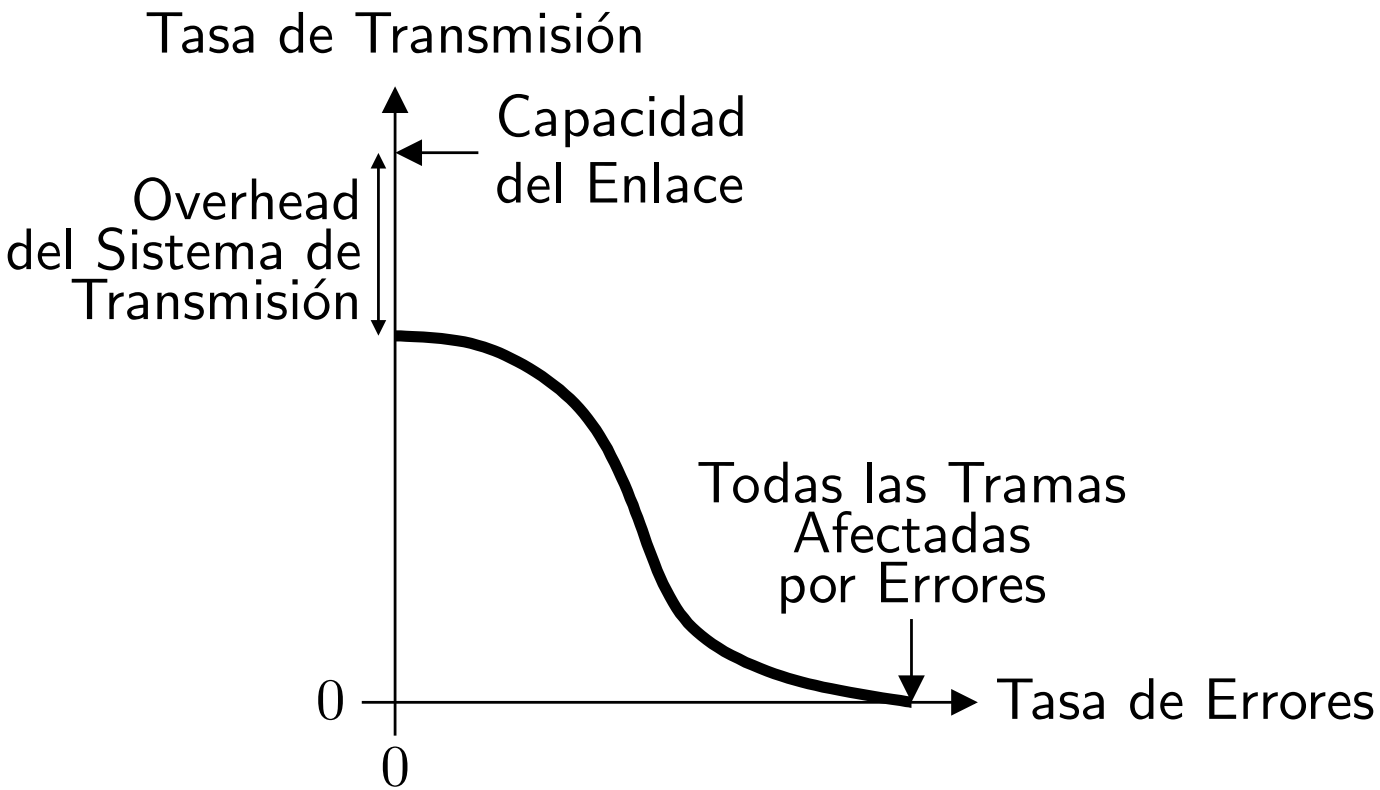



Emisor
Receptor
A0
B1
C2
D3
E4
B1
F5
Aceptar
NAK1
Aceptar
Aceptar
Aceptar
ACK4, Aceptar
$n = 6, l = 12$

13.7. Consideraciones sobre la eficiencia

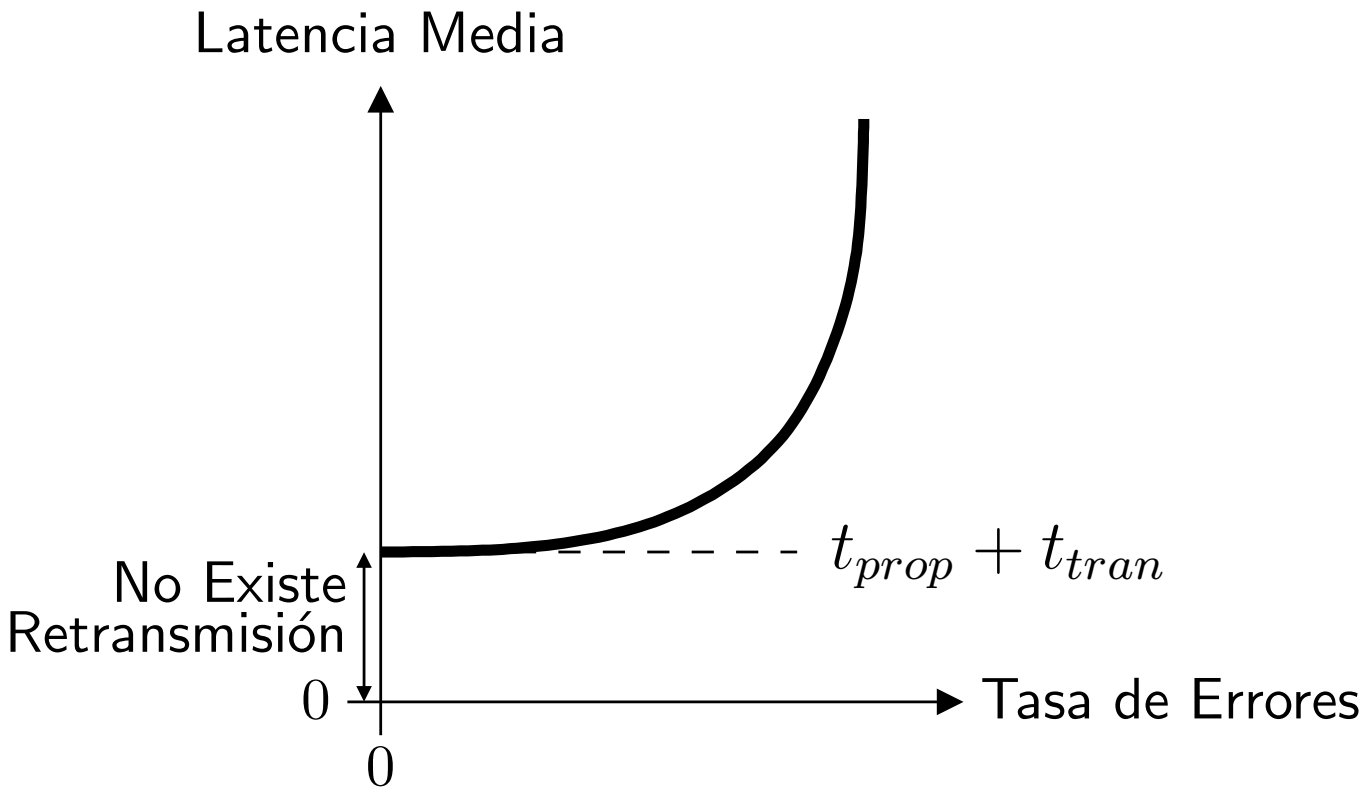
13.7.1. Tasa de transmisión versus tasa de errores

- La tasa de transmisión se degrada al aumentar la tasa de errores [17]:



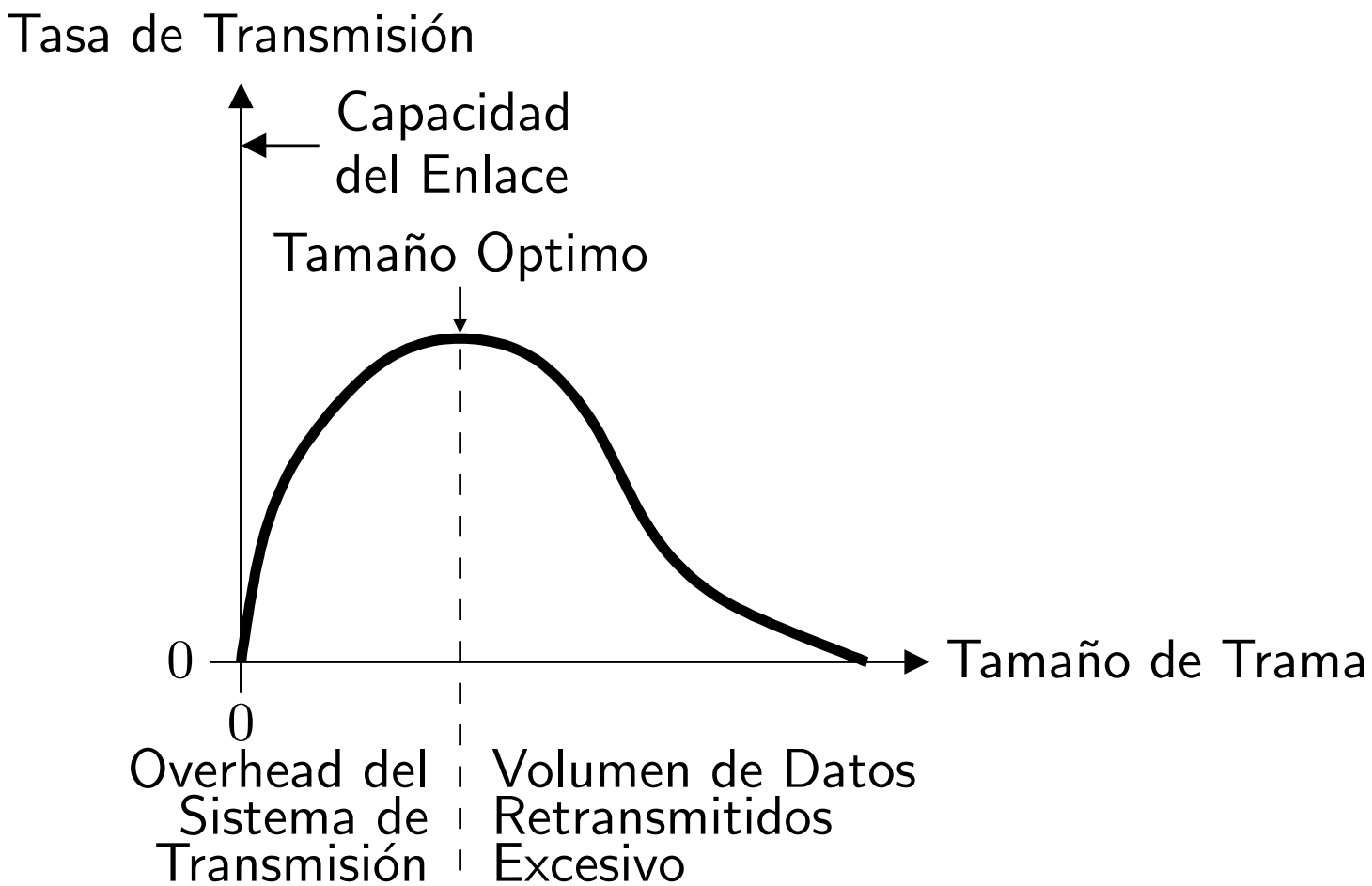
13.7.2. Latencia media versus tasa de errores

- La latencia media aumenta al aumentar la tasa de errores:



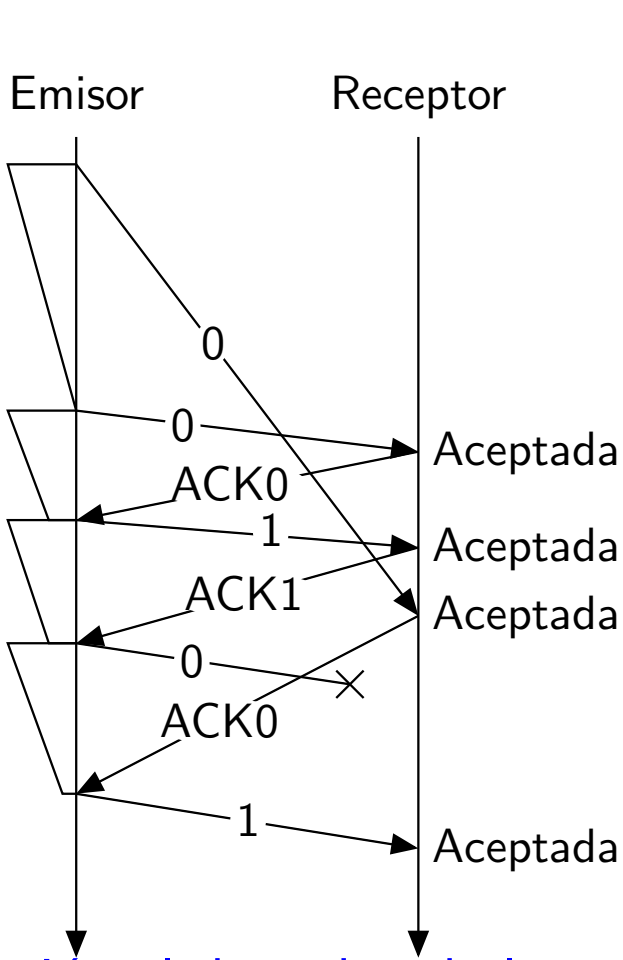
13.7.3. Tasa de transmisión versus tamaño del paquete

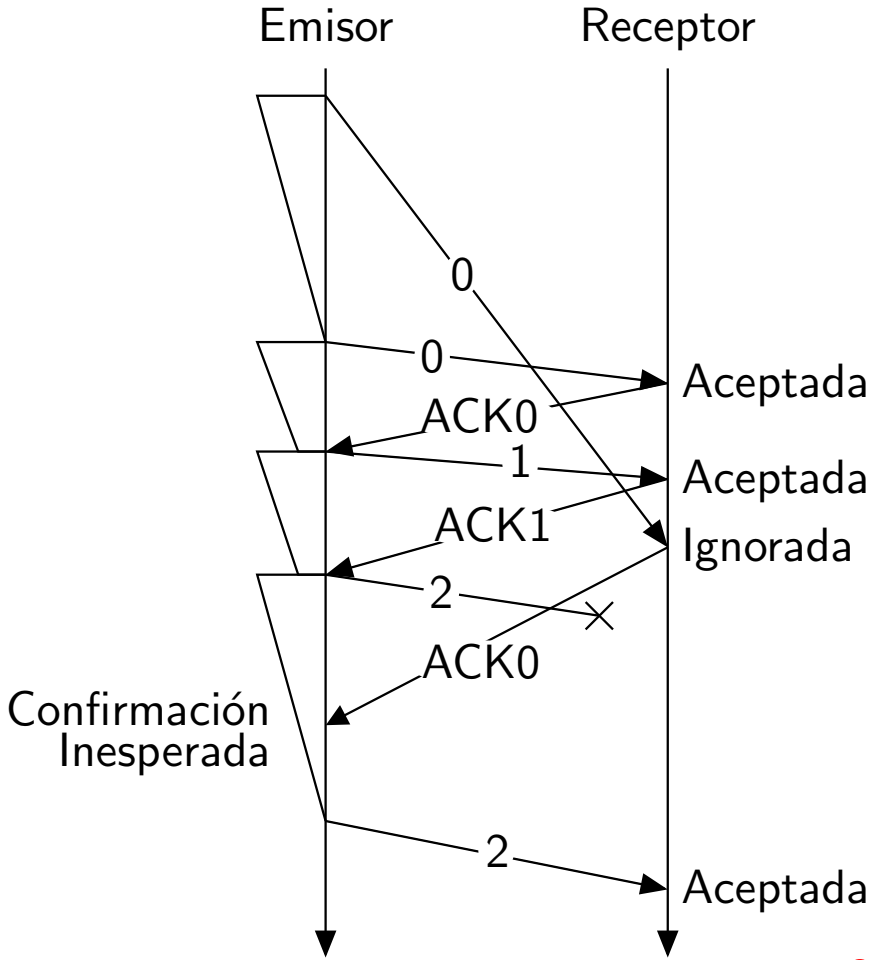
- La tasa de transmisión depende del tamaño del paquete [17]:



13.8. Solución al desorden de los paquetes

- Es necesario utilizar secuencias de conteo suficientemente largas [8]. Por ejemplo, en TCP se utiliza 2^{32}. Para el caso concreto del ejemplo planteado para parada y espera es suficiente contar hasta 3:





Capítulo 14

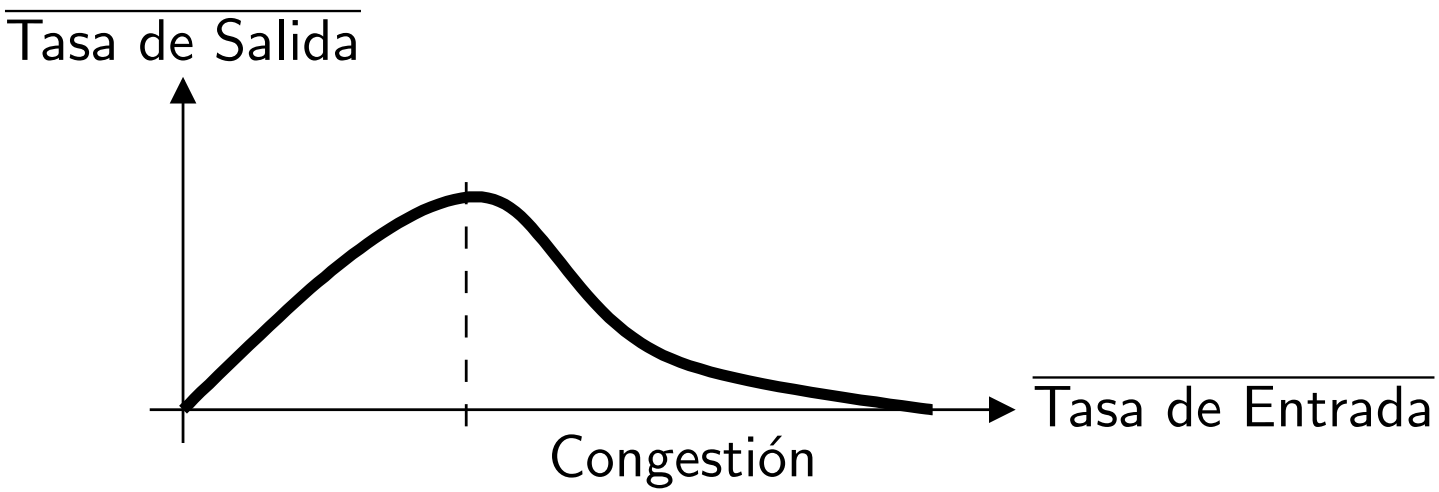
Control del flujo y de la congestión

14.1. Control de flujo

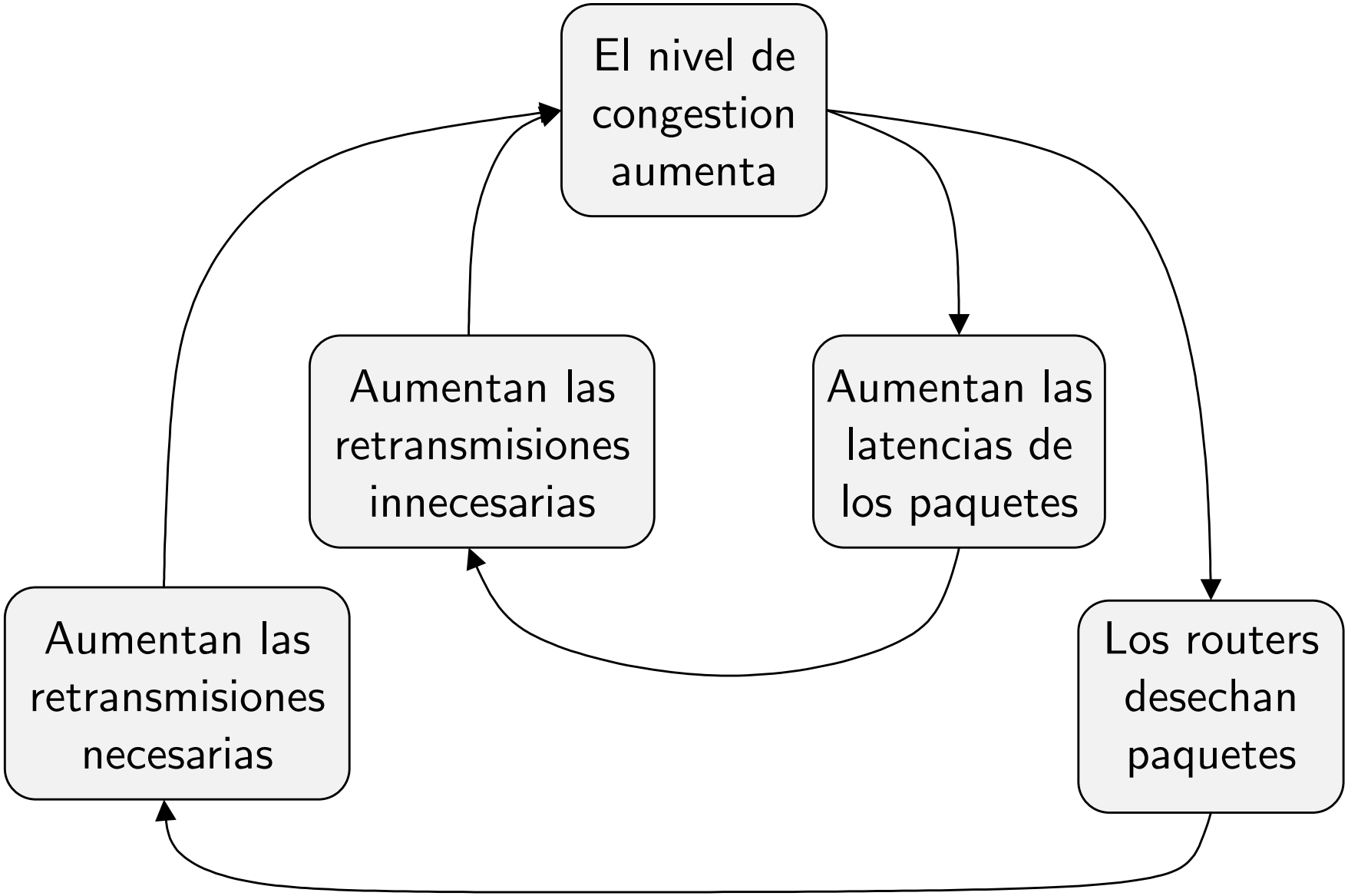
- El control de flujo adecúa la velocidad de transferencia del emisor a la velocidad de procesamiento del receptor [14].
- Como hemos visto, los protocolos ARQ implementan implícitamente el control de flujo porque es el receptor el que marca qué cantidad de datos desea recibir.
- Esto se hace en última medida controlando el número máximo de paquetes que pueden enviarse sin confirmación, en otras palabras, controlando el tamaño de la ventana emisora.

14.2. Control de la congestión

- La congestión es un término que define la carga excesiva de la red de comunicación [14].
- En presencia de la congestión ocurre que la tasa de entrada de datos a la red (en promedio) es superior a la tasa de salida de datos de la red (también en promedio). En la práctica lo que se obtiene es una curva de la forma:



14.3. Causas y costes de la congestión



14.4. ¿Dónde se realiza el control de la congestión?

- Existen dos posibilidades:

  1. **Control de la congestión entre extremos**. La capa de red no proporciona ningún soporte explícito a la capa de transporte para controlar la congestión. De hecho, ésta debe ser inferida por los sistemas finales basándose exclusivamente en la observación del comportamiento de la red (pérdidas y retrasos de paquetes de datos) [14].
  2. **Control de la congestión asistido por la red**. La capa de red (y en concreto los routers) proporcionan una retro-alimentación explícita al emisor sobre el estado de la congestión de la red.

Capítulo 15

El TCP (Transmission Control Protocol)

15.1. Servicios proporcionados

- Protocolo de transporte por excelencia en Internet [24, 4, 14].
- RFC's 793 (principalmente), 854, 1122, 1323, 2018, 2581 y 2988.
- Transferencia fiable de datos (control de errores y de flujo) basado en un protocolo ARQ con pipelining (mezcla de GBN y SR).
- Es orientado a conexión (antes de transmitir ésta se establece).
- Las conexiones son siempre unicast (entre dos procesos). No se puede realizar multicasting.
- Multiplexación: el TCP distingue procesos dentro del mismo host a partir del puerto en el que escuchan y a quién están escuchando.
- Control agresivo de la congestión (por el bien común).
- Full-duplex (comunicación en ambos sentidos al mismo tiempo).
- Los procesos ven byte-streams, no datagramas o segmentos.

15.2. El contexto de trabajo

- Corre encima del IP (capa de red) lo que significa que los paquetes de datos se pueden perder, desordenar, duplicar y llegar con errores.
- Las latencias son muy variables lo que complica conocer la duración de los temporizadores utilizados para conocer cuándo hay que retransmitir los segmentos.
- Se implementa sólo en los sistemas finales. Los routers no distinguen entre paquetes UDP o TCP (ellos ven datagramas, no conexiones).
- Supone que la red no proporciona información acerca de la congestión, lo que significa un ahorro considerable en la complejidad de los dispositivos de nivel 3 (routers) e inferiores.

15.3. Transmisión de datos

15.3.1. El segmento TCP

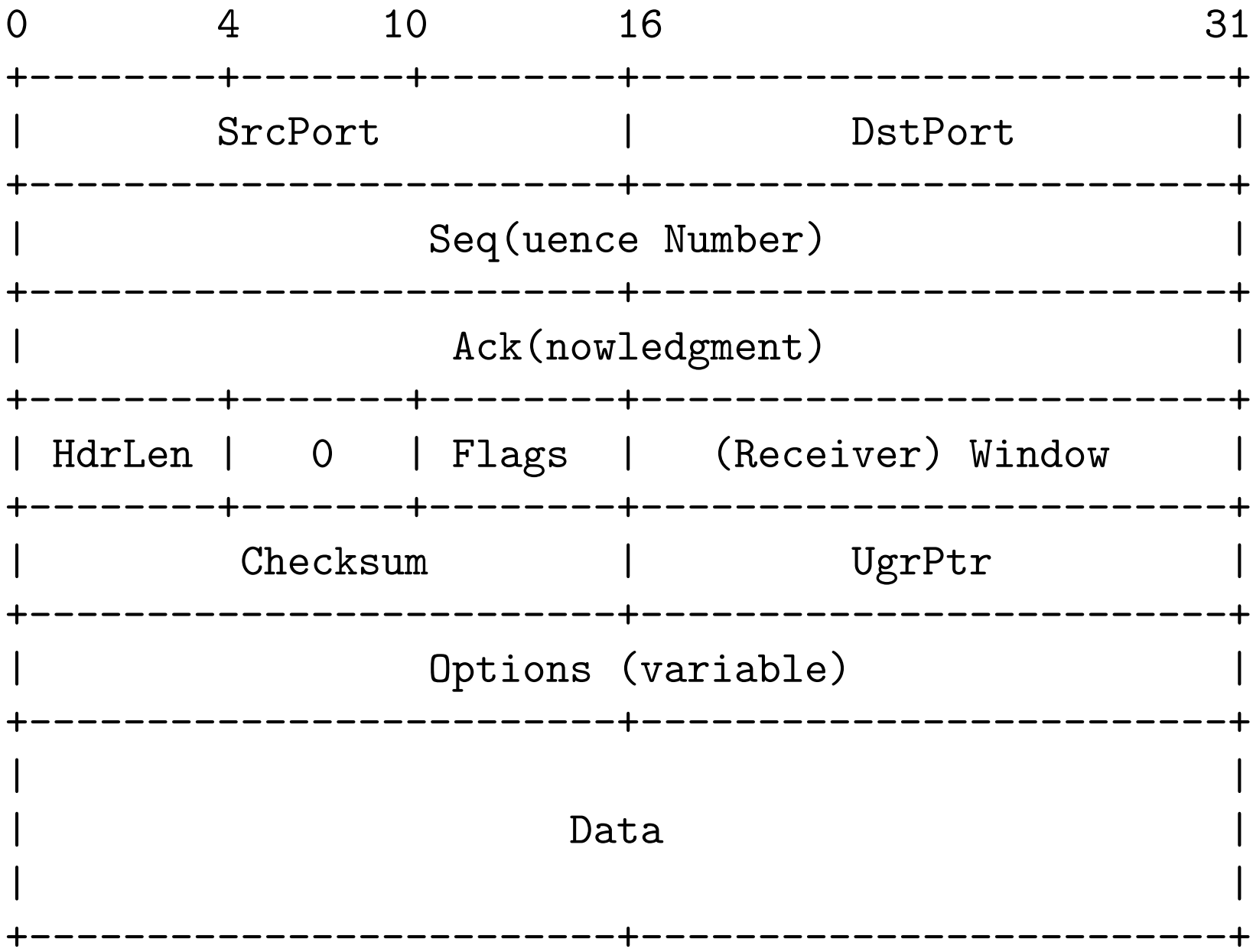
- Aunque las aplicaciones leen y escriben un flujo de bytes, el TCP agrupa los bytes en segmentos que son transmitidos como los datagramas UDP. Esto se hace para minimizar el overhead provocado por las cabeceras de los segmentos (20 bytes como mínimo).

- El tamaño de cada segmento depende de:

  1. El MTU (Maximum Transfer Unit)[1] de la red directamente conectada (en la que está el host). Si el tamaño del segmento es más grande que el MTU, el IP fragmentará los segmentos y si uno se pierde, el TCP tendría que retransmitir todo el segmento a nivel de transporte, no el sub-segmento (paquete) perdido a nivel de IP.
  2. El instante en que el emisor realiza un *push* del flujo (enviar ahora).
  3. Un tiempo máximo de espera.

[1] Tamaño máximo de la trama de datos que el medio físico soporta.

- El formato del segmento TCP es:



`SrcPort`	Puerto del proceso emisor.
`DstPort`	Puerto del proceso receptor.
`Seq`	Indice del primer byte de datos del segmento dentro del flujo de datos.
`Ack`	Siguiente byte esperado.
`Window`	Tamaño de la ventana de recepción del emisor del segmento (núm. de bytes que puede recibir sin enviar un nuevo ACK).
`HdrLen`	Longitud de la cabecera en palabras de 32 bits.
`Flags`	6 bits que pueden representar: SYN = Sincronización de núm. de secuencia. FIN = Fin de la transmisión. RESET = Fin anómalo de la transmisión. PUSH = Segmento enviado mediante un *push*. URG = Datos urgentes (no ignorar en receptor). ACK = Segmento contiene `Ack`.
`Checksum`	De la cabecera, pseudo-cabecera y datos. Obligatorio.
`UgrPtr`	Punto donde se finalizan los datos urgentes.
`Options`	MSS, marca temporal y factor de escala.
`Data`	Datos transmitidos.

- La pseudo-cabecera en TCP es idéntica a la del UDP, excepto en que el campo `PROTO` $= 6$.

15.3.2. EL proceso de desmultiplexación

- Está basada en conexiones (par de puntos extremos), no en puertos.
- El TCP diferencia a un proceso dentro de un host a partir del par de puntos extremos

  (punto extremo origen, punto extremo destino)

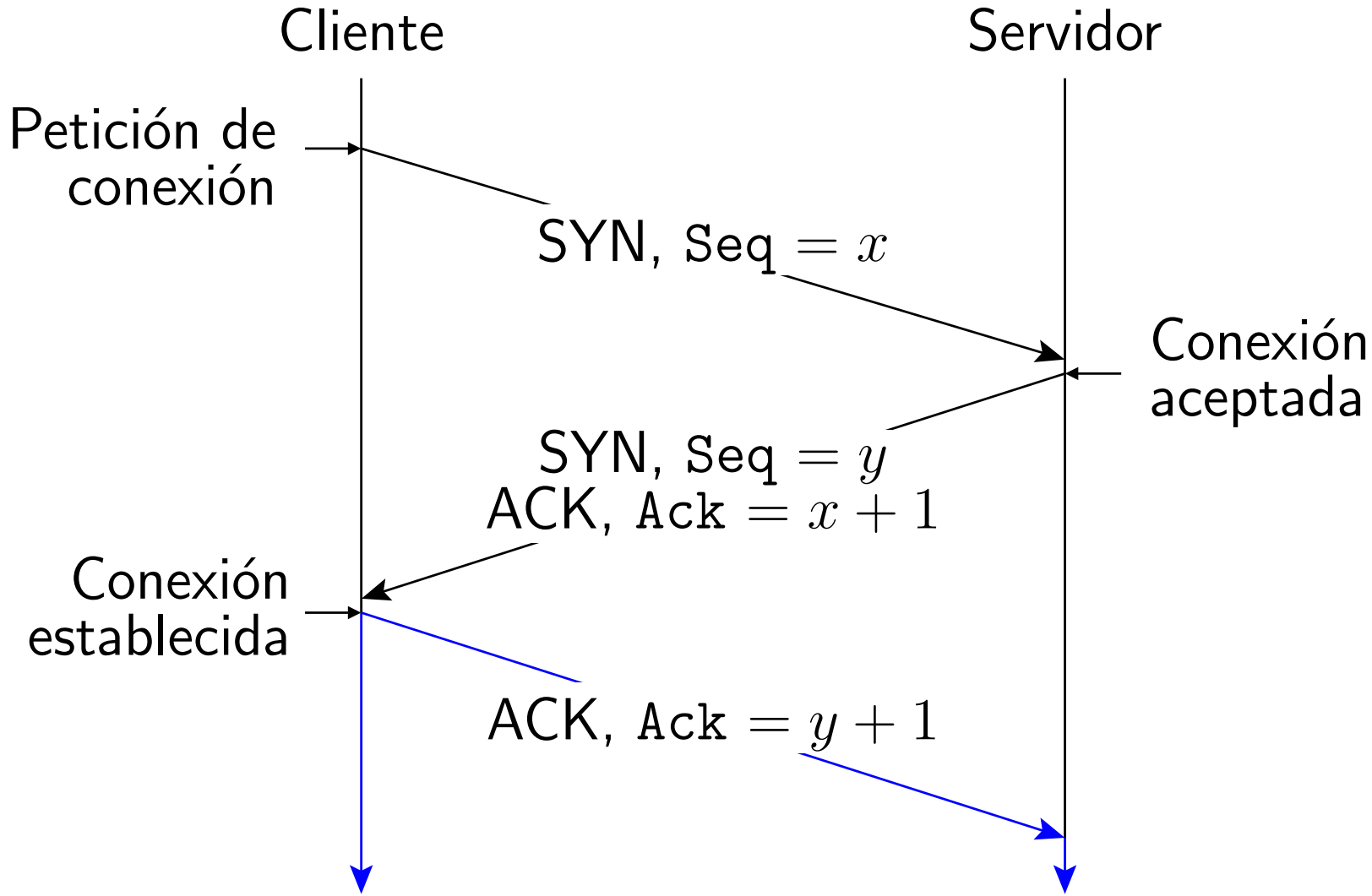
  donde un punto extremo está formado por un par

  (IP del host, número de puerto).

- Por tanto, dos procesos distintos en un mismo host que usan en un mismo puerto pueden estar conectados a dos procesos distintos en otro host si estos utilizan puertos de salida diferentes (ver el progama `netstat` de Unix).

15.3.3. Establecimiento de la conexión

Se utiliza el Algoritmo *Three-Way Handshake*:



- x e y, los números de secuencia inciales, suelen ser un números aleatorios. Motivos:

  1. Para minimizar la probabilidad de que un segmento de una conexión anterior ya terminada y que se encuentra viajando por la red sea tomado erróneamente por un segmento válido de una conexión posterior entre los mismos puntos extremos [14].
  2. Para evitar el ataque del "número de secuencia" [18] que consiste en que un cliente "ilegal" puede suplantar a un cliente "legal" averiguando los números de secuencia de los segmentos[2], usando su IP y su puerto de salida. Tras la suplantación el cliente "legal" podría acceder a datos restringidos o cerrar la conexión con el servidor.

- Los dos primeros segmentos son de control, ninguno puede transportar datos. El tercero sí puede transportar datos [15]. A partir del tiempo

[2] Lo que seguro es más sencillo si los números de secuencia iniciales estuvieran predeterminados.

en línea azul, los procesos pueden emitir y recibir segmentos normales con datos.

15.3.4. El tamaño máximo de los segmentos

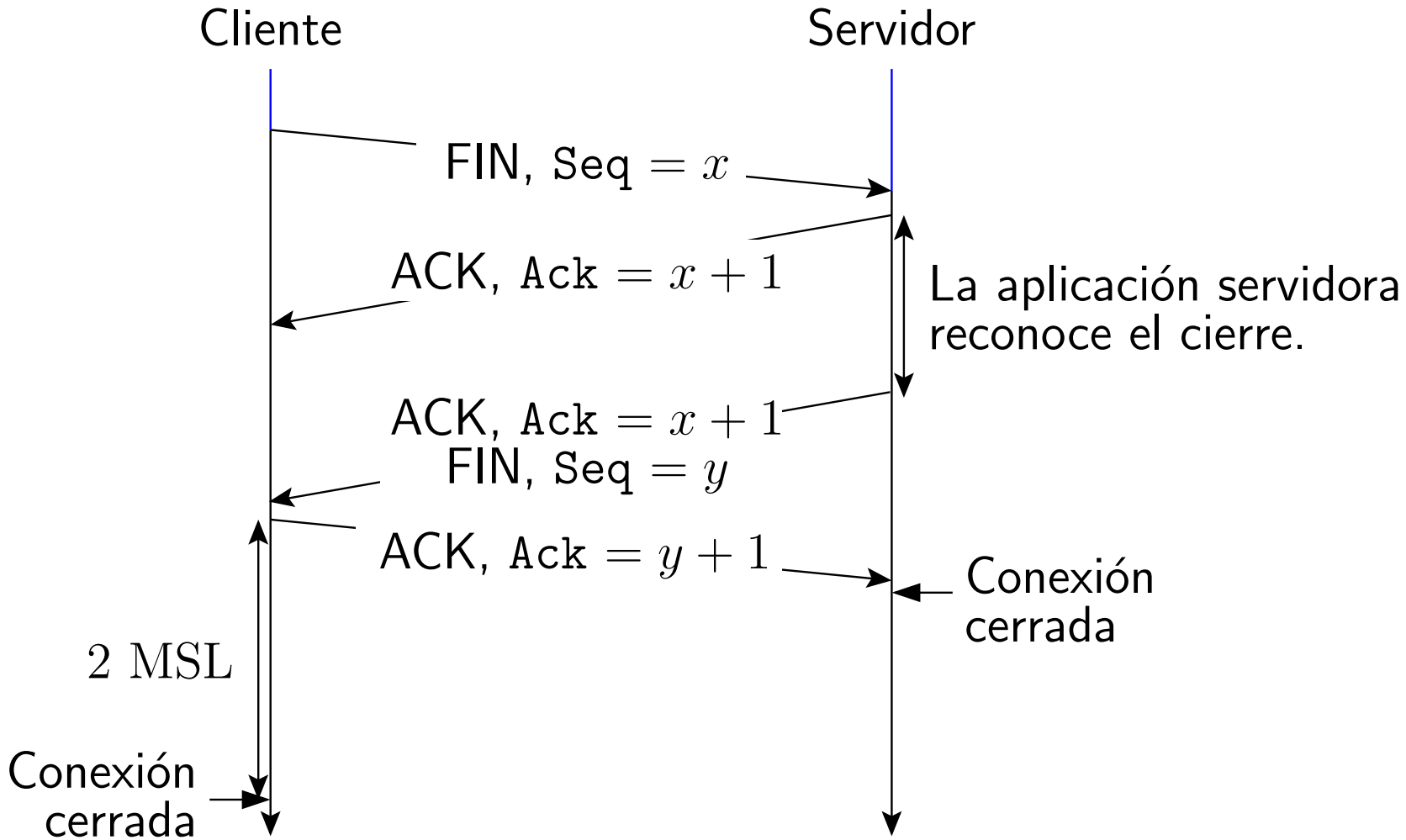
- Los extremos de una transmisión TCP necesitan conocer de antemano, a partir del establecimiento de la conexión, el tamaño máximo de los payloads (la parte con datos) de los segmentos transmitidos [14]. A este valor se le llama tamaño máximo de segmento o MSS (Maximum Segment Size).
- Dicho valor se establece en el campo `Options` y depende normalmente del MTU de la red asociada. Si el MSS más el tamaño de las correspondientes cabeceras es mayor que el MTU, el IP fragmentará el segmento. La pérdida (por parte del IP) de uno de los fragmentos significa que el TCP deberá reenviar todo el segmento.
- Por tanto, si el segmento es demasiado grande, la cantidad de datos retransmitidos por errores de transmisión aumenta y disminuye la tasa efectiva de transmisión.

15.3.5. Transmisión de datos urgentes

- Existen situaciones donde los datos deben entregarse a la aplicación receptora lo antes posible. TCP indica esto en sus segmentos activando el flag URG.
- El tratamiento de los segmentos urgentes se produce sólo entre las capas de transporte del emisor y del receptor. El IP es ajeno y trata por igual a todos los segmentos, independientemente de si el flag URG está activado.
- Cuando dicho flag está activado, `UrgPtr` indica hasta qué parte de los datos a partir del comienzo de los datos son de carácter urgente.
- Cuando llega un segmento con el flag UGR activado hasta el receptor, la parte urgente de éste es entregado al proceso receptor lo antes posible.

15.3.6. Cierre de la conexión

- Generalmente es el cliente el que cierra la conexión. Esto se hace con el Algoritmo *Four-Way Handshake*:


Cliente
Servidor
FIN, Seq = x
ACK, Ack = $x + 1$
La aplicación servidora reconoce el cierre.
ACK, Ack = $x + 1$
FIN, Seq = y
ACK, Ack = $y + 1$
Conexión cerrada
2 MSL
Conexión cerrada

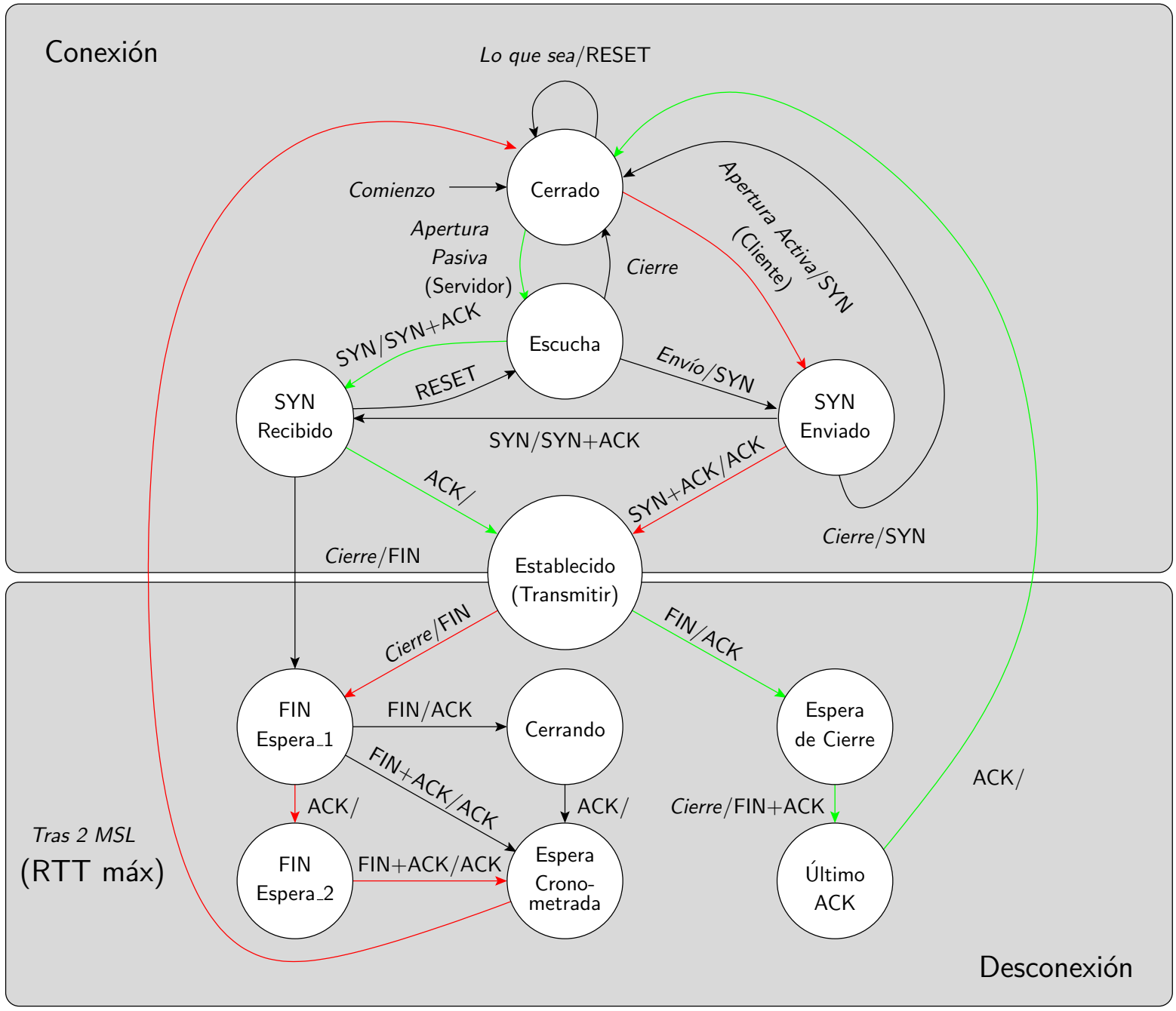
- El cliente, tras la emisión del segundo ACK espera típicamente 2 MSL[3] (2×60 segundos en la mayoría de las implementaciones [22, 5]) que es intervalo de tiempo en el cual podría recibirse otro ACK+FIN del servidor si el ACK del ciente se perdiera y no llegara al servidor.

  Si esto último ocurriera, el servidor reenviaría el ACK+FIN y si este se retrasara lo suficiente, otra aplicación en el host cliente podría establecer una nueva conexión con la aplicación servidora, usando el mismo puerto de salida. Si a continuación llegara el ACK+FIN retrasado, dicha conexión se cerraría puesto que el cliente considera que el servidor desea cerrar la conexión [22].

[3] MSL o Maximun Segment Lifetime es el tiempo máximo estimado de vida de un paquete IP, y por lo tanto de un segmento, en Internet.

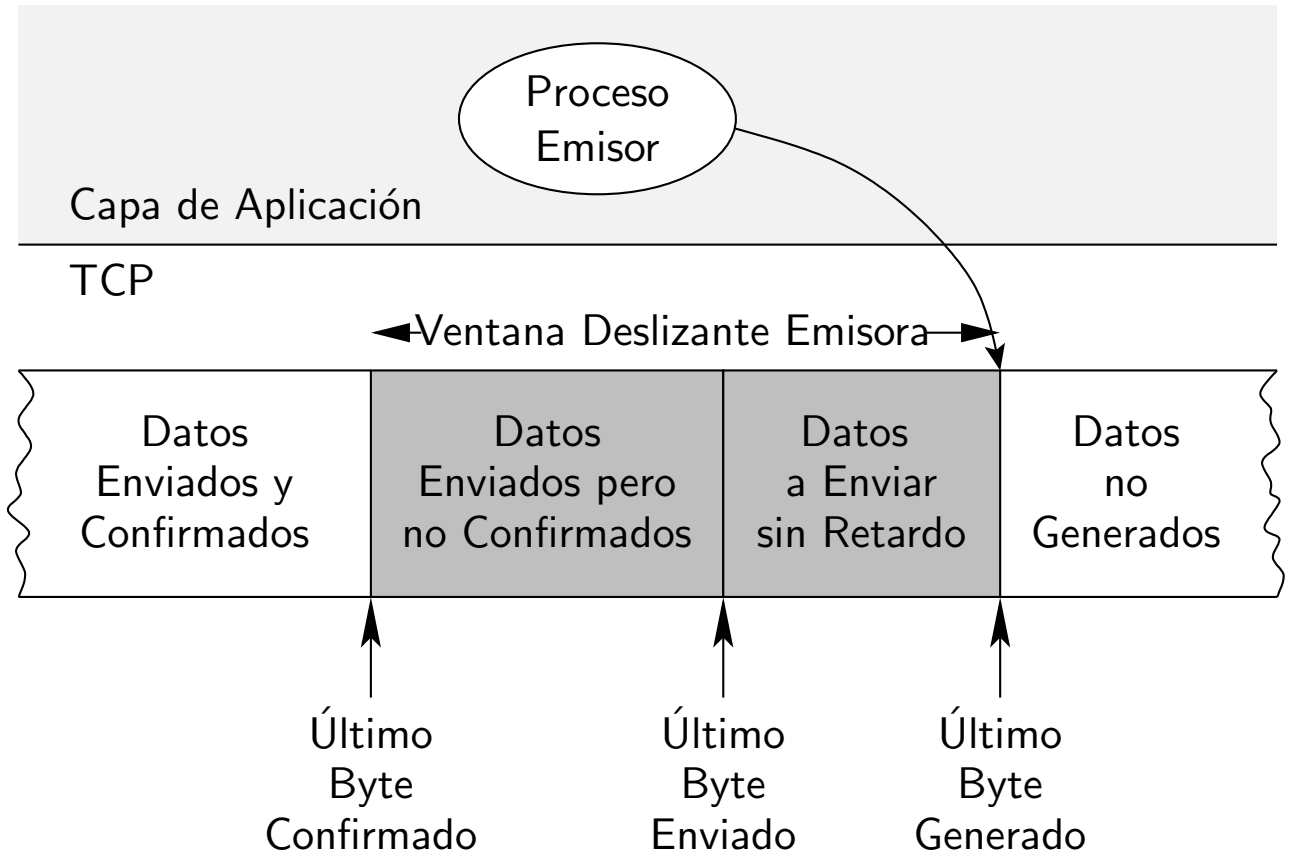
15.3.7. El diagrama de estados

- Frecuentemente se representa mediante una máquina de estados finita donde los nodos representan estados y las transiciones se etiquetan mediante *evento*/*acción*.
- Los eventos puede estar provocados por la llegada de nuevos segmentos con información de control (SYN, FIN, etc.) o porque la aplicación local realiza una llamada al TCP (*Apertura Pasiva*, *Cierre*, etc.).
- No siempre que se produzca un evento necesariamente se genera una salida. En este caso sólo cambiamos de estado.
- En rojo aparecen las transisiones normalmente seguidas por el cliente y en verde las seguidas por el servidor.

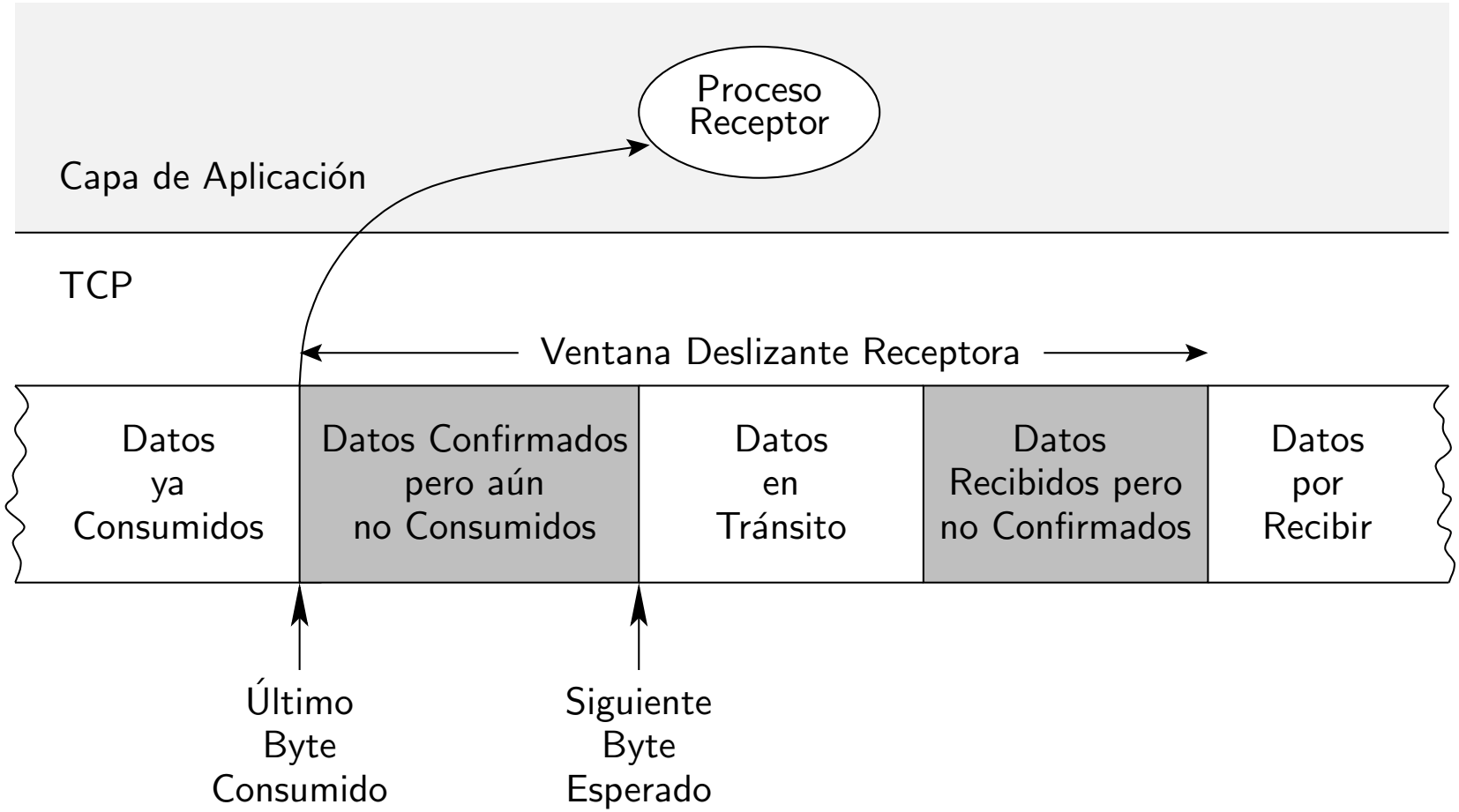

Conexión
Lo que sea/RESET
Comienzo
Cerrado
Apertura
Pasiva
(Servidor)
Cierre
Apertura Activa/SYN
(Ciente)
SYN/SYN+ACK
Escucha
Envío/SYN
RESET
SYN
Recibido
SYN
Enviado
SYN/SYN+ACK
ACK/
SYN+ACK/ACK
Cierre/SYN
Cierre/FIN
Establecido
(Transmitir)
Cierre/FIN
FIN/ACK
FIN
Espera_1
FIN/ACK
Cerrando
Espera
de Cierre
FIN+ACK/ACK
ACK/
ACK/
Cierre/FIN+ACK
ACK/
Tras 2 MSL
(RTT máx)
FIN
Espera_2
FIN+ACK/ACK
Espera
Crono-
metrada
Último
ACK
Desconexión

15.4. Control de flujo y de errores

- El TCP utiliza 2 colas circulares (ventanas deslizantes organizadas en bytes) en cada host, una para transmitir y otra para recibir.
- La estructura de la cola de emisión es la siguiente:



- La estructura de la cola de recepción es:



- El control de flujo lo realiza el receptor controlando el tamaño de la ventana emisora del emisor (campo `Window`).

- Cuando la ventana emisora del emisor se hace 0, el proceso emisor se bloquea, lo que significa que el receptor no va a recibir más datos, al menos temporalmente.[4] Como el emisor no recibe un segmento de control del receptor si éste no le envía un segmento con `Window` > 0, el emisor periódicamente trata de enviarle 1 byte de datos al receptor. Mientras el receptor no puede recibir más datos, un segmento con `Window` $= 0$ es enviado. Sin embargo, cuando el receptor puede recibir datos, este campo es > 0 y la ventana del emisor es ampliada.

[4] Excepto si llega un segmento con su URG activo.

15.4.1. El tamaño de las ventanas

- Cuando el factor de escala[5] del campo `Options` $= 1$, el tamaño máximo de la ventana del emisor es de 64 KB. Este límite es demasiado bajo para la mayoría de los enlaces de media y larga distancia (RTT ≥ 100 ms) [22]:

Enlace	Capacidad	100 ms × Capacidad
T1	1,5 Mbps	18 KB
Ethernet	10 Mbps	122 KB
T3	45 Mbps	549 KB
Fast Ethernet	100 Mbps	1,2 MB
STS-3	155 Mbps	1,8 MB
STS-12	622 Mbps	7,4 MB
STS-24	1,2 Gbps	14,8 MB

- El receptor puede utilizar el factor de escala para aumentar hasta 2^{32} bytes el tamaño de la ventana del emisor.

[5] Expresado siempre como una potencia de 2.

15.4.2. El tamaño de los números de secuencia

- TCP utiliza números de secuencia de 32 bits y el tamaño máximo de las ventanas del emisor y del receptor es de $2^{16} \times 2^{16} = 2^{32}$ bytes. El primer factor 2^{16} está determinado por el campo `Window` (de 16 bits) y el segundo por un factor de escala presente en el campo `Options` (de 8 bits, aunque sólo puede llegar a valer 16). Nótese que estando TCP basado en el protocolo ARQ con pipelining, excepto en el caso extremo se cumple siempre que la longitud de la secuencia de conteo es siempre superior al tamaño máximo de la ventana emisora.

- Por desgracia esta condición no es suficiente, pudiéndose dar el caso de que si la red retrasa suficientemente un segmento, éste puede "suplantar" a otro en un momento posterior de dicha u otra transmisión. Además, este factor aumenta proporcionalmente con la tasa de transmisión de la red [14]. En otras palabras, teniendo en cuenta que el MSL $= 60$ segundos, 32 bits son suficientes para la mayoría de las situaciones, pero no para todas [22]:

Enlace	Capacidad	Tiempo hasta *wrap-around*
T1	1,5 Mbps	6,4 horas
Ethernet	10 Mbps	57 minutos
T3	45 Mbps	13 minutos
Fast Ethernet	100 Mbps	6 minutos
STS-3	155 Mbps	4 minutos
STS-12	622 Mbps	55 segundos
STS-24	1,2 Gbps	28 segundos

Wrap-around = Situación que se produce cuando el número de secuencia sobrepasa el límite $2^{32} - 1$ y comienza de nuevo por el principio.

- Para solucionar el problema, en el campo `Options` de los segmentos TCP se coloca una marca de tiempo. De esta forma el receptor puede saber cuál de los dos segmentos que tienen el mismo número de secuencia es el más viejo.

15.4.3. Retransmisión rápida

- El TCP no usa NAK's. Sin embargo, cuando se pierde un segmento el emisor puede percatarse de ello antes de que expire su temporizador si recibe uno o más ACK's duplicados, es decir, cuando recibe un reconocimiento para un segmento que ya había sido reconocido.
- En la práctica, cuando se reciben 3 ACK's del mismo segmento no se espera a que expire el temporizador y se realiza una retransmisión rápida del segmento [14]. Los diseñadores del TCP estimaron que esperar sólo a una duplicación del ACK podría fallar a la hora de estimar una pérdida cuando en realidad lo que ha ocurrido es que un segmento ha adelantado a otro.

15.4.4. El síndrome de la ventana tonta

- Se genera con frecuencia cuando conectamos un emisor rápido a un receptor lento.
- El receptor consume los datos byte a byte, y con cada uno de ellos envía un segmento de confirmación al emisor. El resultado es que los segmentos enviados sólo contienen un byte de datos lo que consume muchos recursos de la red.[6]
- La misma situación aparece si el emisor es demasiado ansioso y no espera a tener suficientes datos antes de enviarlos.
- Para prevenirlo:
  1. El receptor tratará de evitar el anunciar ventanas pequeñas. En la práctica espera a recibir

$$\text{mín}(\texttt{Window}/2, \text{MSS}) \text{ bytes},$$

[6] La eficiencia desde el punto de vista del TCP/IP sería de 1/41, 20 bytes de la cabecera TCP y otros 20 de la IP.

donde `Window` es el tamaño de la ventana del receptor. Esto implica que los ACK's pueden retrasarse indefinidamente lo que supone un problema para el emisor (para el cálculo del time-out) y para la red (por las retransmisiones innecesarias debidos a dichos retrasos). En la práctica el TCP limita el retraso máximo de un ACK a 500 ms.

2. El emisor tratará de evitar el usar segmentos pequeños (*tinygrams*), es decir, acumulará datos hasta generar segmentos iguales al MSS mientras no reciba un ACK. Así, si la aplicación emisora es el cuello de botella y la aplicación receptora no está ansiosa (no envía ACK's constantemente), el tamaño de los segmentos será grande. A este procedimiento se le conoce como Algoritmo de Nagle.

15.5. Control de la congestión

- Si el TCP percibe que la red se está congestionando disminuye la tasa de transmisión y viceversa.
- Un emisor generalmente no conoce dónde ni por qué se está produciendo la congestión. Simplemente experimenta un incremento en los RTT's y tal vez reciba algún paquete ICMP para informar de esta situación. Ante ello, el TCP debe reducir la tasa de transmisión [14].
- Si la congestión se hace más severa los routers comienzan a descartar paquetes. El TCP debe percatarse de este problema y no retransmitirlos hasta que la congestión cese.
- Por este motivo, el tamaño de la ventana emisora no sólo depende de lo que indica el receptor (control del flujo), sino que también va a tener en cuenta el nivel de congestión.

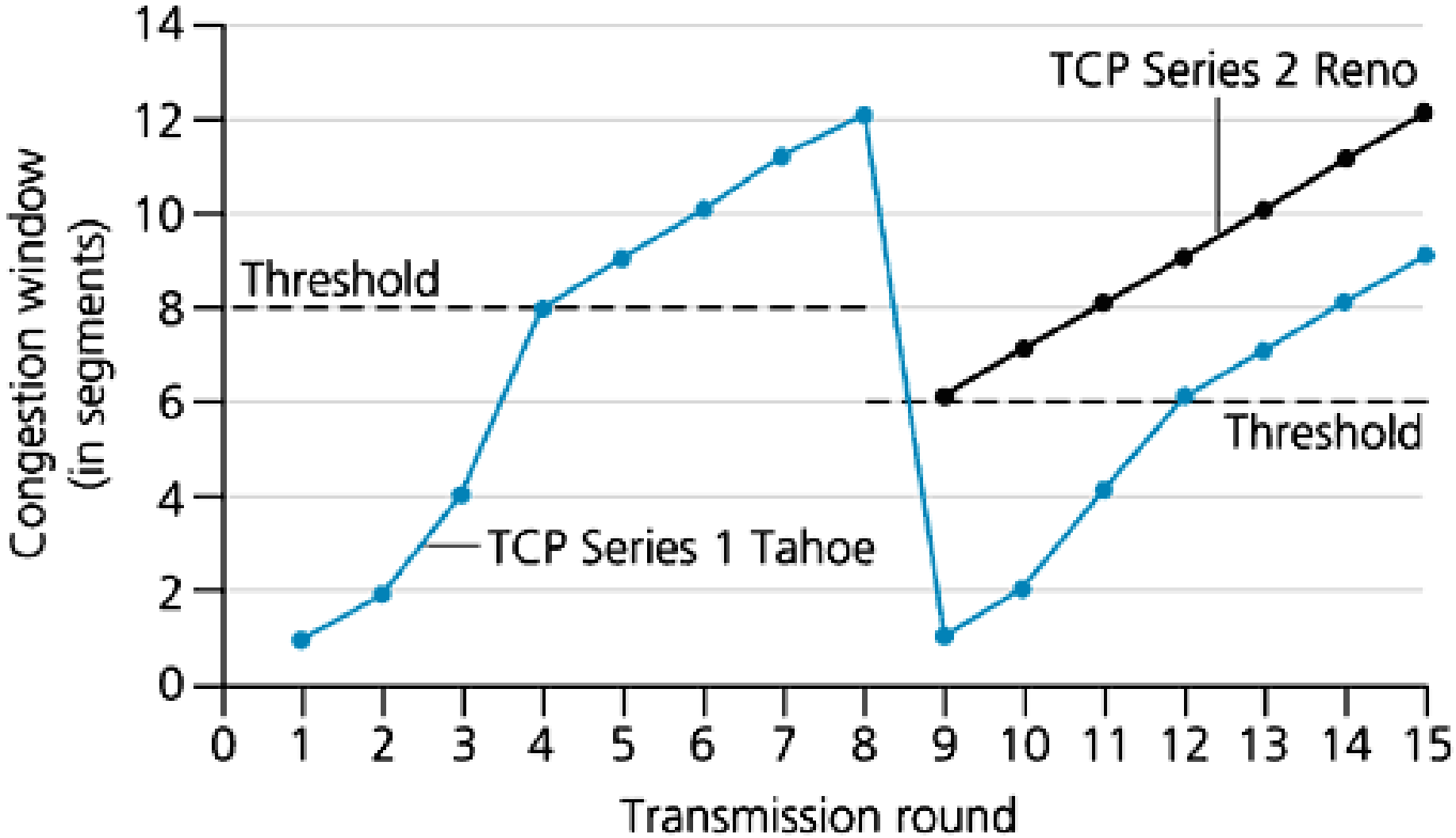
- En la práctica, el tamaño de la ventana emisora $SenderWindowSize$ va a ser igual al menor del tamaño indicado por el receptor `Window` y un valor $CongestionWindowSize$ impuesto por el nivel de congestión de la red, es decir,

$$SenderWindowSize = \text{mín}(\texttt{Window}, CongestionWindowSize).$$

El valor $CongestionWindowSize$ se calcula siguiendo las siguientes reglas:

1. **Modo de arranque** (relativamente) **lento** en el que se duplica $CongestionWindowSize$ con cada ACK recibido. Este modo se utiliza desde que desaparece la situación de congestión hasta que se alcanza la mitad del anterior valor de $CongestionWindowSize$ (que en la gráfica se llama Threshold).
2. **Modo de prevención de la congestión** en el que $CongestionWindowSize$ crece en un MSS por cada ACK recibido. Este modo perdura hasta que se pierde un segmento[4].

3. Cuando se detecta la pérdida de un segmento por triple ACK recibido, entonces $CongestionWindowSize \leftarrow CongestionWindowSize/2$ (este hecho no se muestra en la gráfica).
4. Cuando se detecta la pérdida de un segmento porque expira su temporizador, entonces $CongestionWindowSize \leftarrow 1$ (este hecho se muestra en la gráfica).


Congestion window
(in segments)
14
12
10
8
6
4
2
0
Threshold
TCP Series 2 Reno
Threshold
TCP Series 1 Tahoe
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Transmission round

15.5.1. El tamaño de los time-outs

- Cada vez que se envía un segmento, el TCP arranca un temporizador y espera el correspondiente ACK. Si se termina el tiempo antes de que se confirme positivamente el segmento (recuérdese que no se utilizan confirmaciones negativas), el TCP asume que dicho segmento se perdió o corrompió, y lo retransmite.
- El problema radica en que las redes IP poseen latencias muy variables debido a su tamaño, heterogeneidad, variación de la carga, nivel de congestión, etc. Por este motivo el TCP utiliza un algoritmo adaptativo para calcular los time-outs.

15.5.1.1. El algoritmo original

Estima el RTT promedio como

$$EstimatedRTT \leftarrow \alpha \times EstimatedRTT + (1 - \alpha) \times SampleRTT,$$

donde:

$EstimatedRTT$ es la estimación del RTT,

$SampleRTT$ es la medida del último RTT (medido como la diferencia en tiempo entre el instante en que se envía el segmento hasta que se recibe su confirmación) y

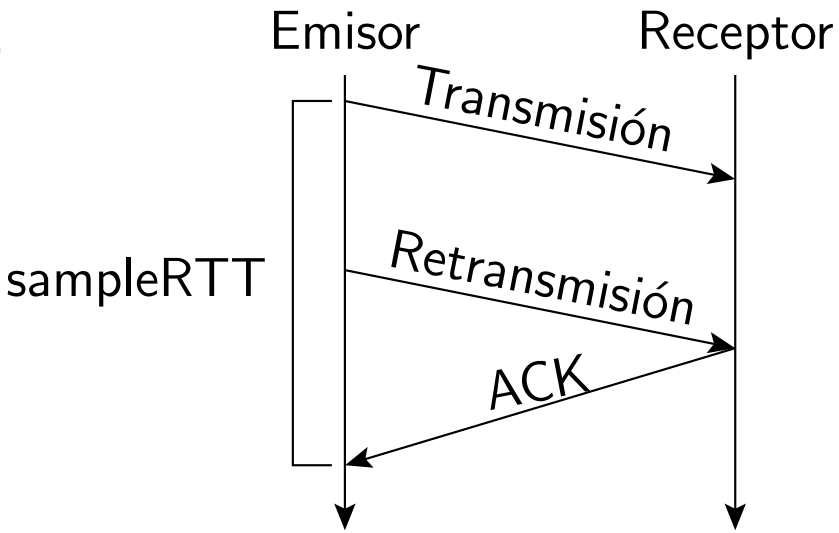
α es un número entre 0 y 1 de tal forma que si $\alpha \approx 0$ entonces tiene mayor peso la última medida del RTT, y si $\alpha \approx 1$ entonces $EstimatedRTT$ es menos dependiente de $SampleRTT$. La especificación original del TCP recomienda que $0{,}8 \leq \alpha \leq 0{,}9$.

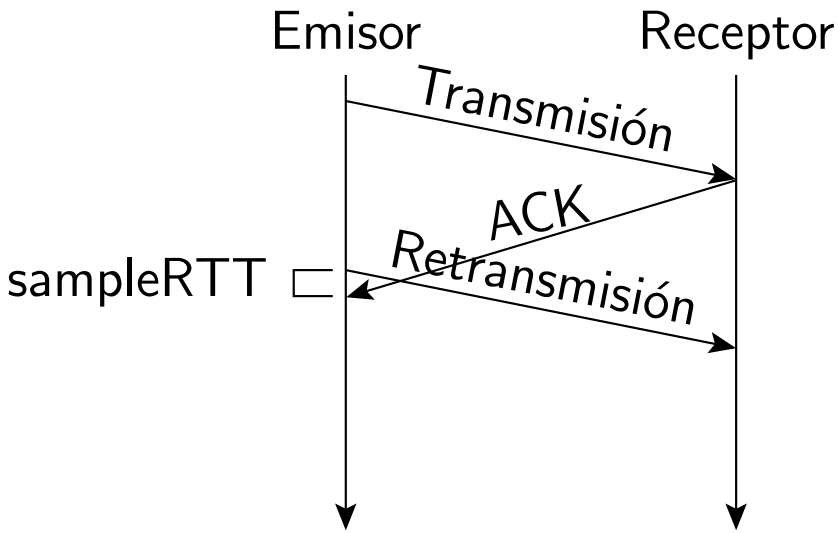
Finalmente, la estimación (promedio) para el time-out es

$$TimeOut \leftarrow 2 \times EstimatedRTT.$$

15.5.1.2. El Algoritmo de Karn/Partridge

El algoritmo original no funciona correctamente cuando ocurre una retransmisión. En esta situación, el emisor no puede saber si el ACK recibido se corresponde con la primera o la segunda transmisión (problema de la ambigüedad del ACK). Si supone que es con la primera y no ocurre así, el $SampleRTT$ medido sería demasiado largo. Si supone que es con la segunda y esto no es cierto, el cálculo es demasiado corto. Gráficamente:





Ambos errores son perjudiciales para el cálculo del $TimeOut$:

- Si el $SampleRTT$ (y por lo tanto el $TimeOut$) es más grande de lo que debiera ser, las retransmisiones ocurrirán con un periodo superior. Así, el siguiente $SampleRTT$ tras una retransmisión tenderá a crecer, y así sucesivamente. En consecuencia, un $TimeOut$ excesivamente grande provoca a la larga una infrautilización de los enlaces.
- Si el $SampleRTT$ (y por lo tanto el $TimeOut$) es erróneamente pequeño, el emisor retransmitirá innecesariamente antes de recibir el correspondiente ACK. Esto provocará de nuevo que el SampleRTT sea erróneamente pequeño y así sucesivamente hasta llegar a una situación de equilibrio donde en promedio cada segmento se retransmite 1 vez. Por tanto, un $TimeOut$ excesivamente pequeño incrementará la congestión de los enlaces.

La solución:

> *Ignorar los RTT's de los segmentos retransmitidos para computar el time-out*.

Estimar el RTT sólo para los segmentos que no son retransmitidos acarrea un nuevo problema. Supongamos que debido a un aumento en la carga de la red o de la congestión, el RTT aumenta por encima del time-out, lo que provoca una retransmisión. Como el TCP va a ignorar el RTT de los segmentos retransmitidos, nunca va a actualizar la estimación para el RTT mientras las latencias sigan siendo altas y el ciclo continuará. Por este motivo, además:

> *Cada vez que se retransmite se dobla el time-out, hasta alcanzar $2 \times MSL$ [4].*

El motivo de este aumento exponencial del time-out obedece a que normalmente un aumento en las latencias se debe a problemas de congestión y ante esta situación lo mejor es disminuir rápidamente la frecuencia de las retransmisiones.

15.5.1.3. El Algoritmo de Jacobson/Karels

Por desgracia, la mejora del algoritmo básico realizada por Karn y Partridge no funciona cuando los niveles de congestión son altos, ya que las latencias son muy variables. En esta situación es muy importante no colapsar aún más la red usando time-outs demasiado bajos.

La mejora propuesta por Jacobson y Karels consiste en tener en cuenta además la variación de los RTT's, no sólo su media. Intuitivamente, si la varianza es baja, entonces $EstimatedRTT$ es más fiable. Por otra parte, una varianza alta significa que $EstimatedRTT$ no debe ser considerado con tanto peso a la hora de calcular $TimeOut$. De esta forma, el cálculo actual del time-out en el TCP es

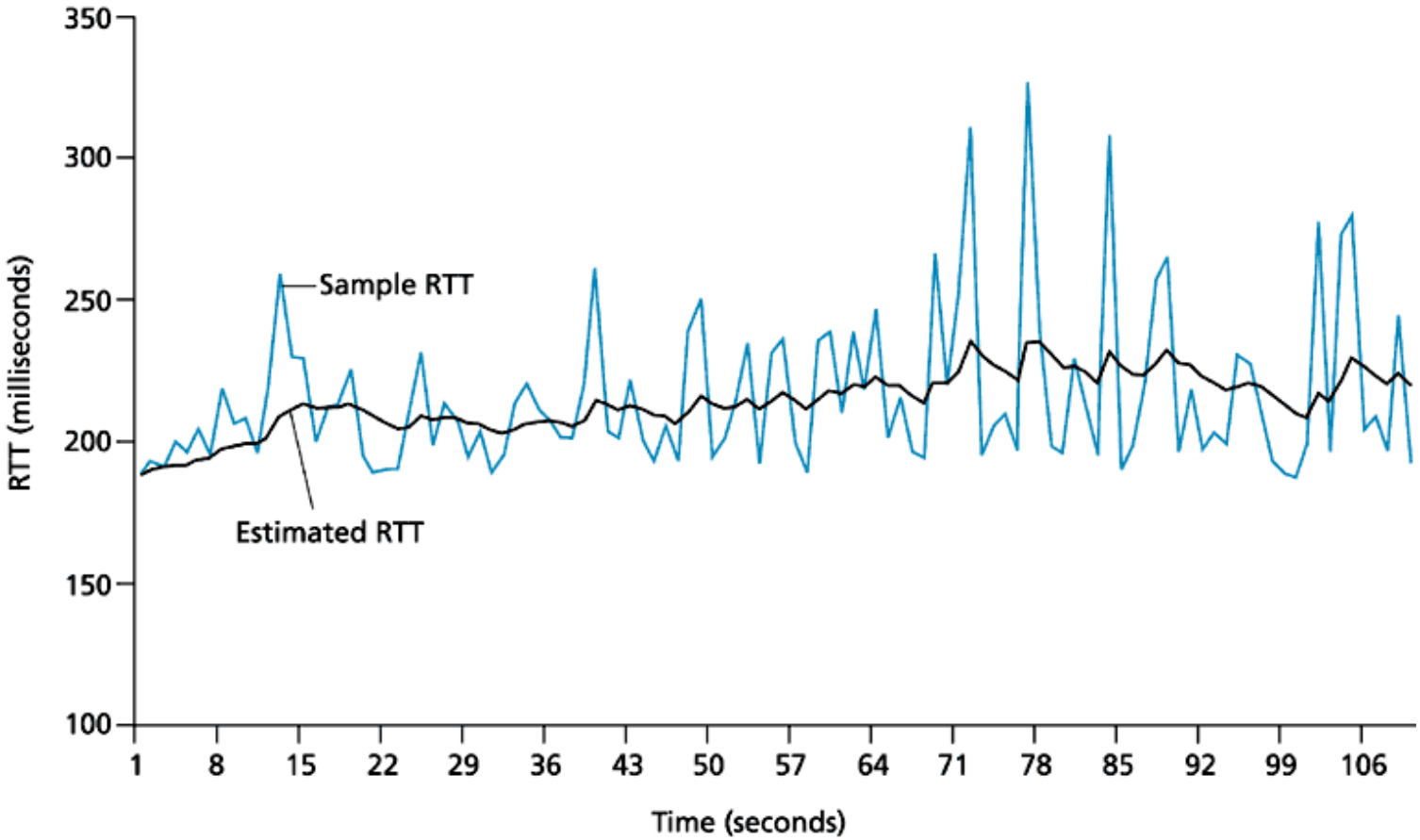
$$\begin{aligned} Diff &\leftarrow SampleRTT - EstimatedRTT, \\ EstimatedRTT &\leftarrow EstimatedRTT + \delta \times Diff, \\ DevRTT &\leftarrow DevRTT + \rho \times (|Diff| - DevRTT), \\ TimeOut &\leftarrow EstimatedRTT + 4 \times DevRTT, \end{aligned}$$

donde:

- $0 \leq \delta \leq 1$ es un factor que mide lo que $Diff$ afecta al nuevo cálculo

del RTT y

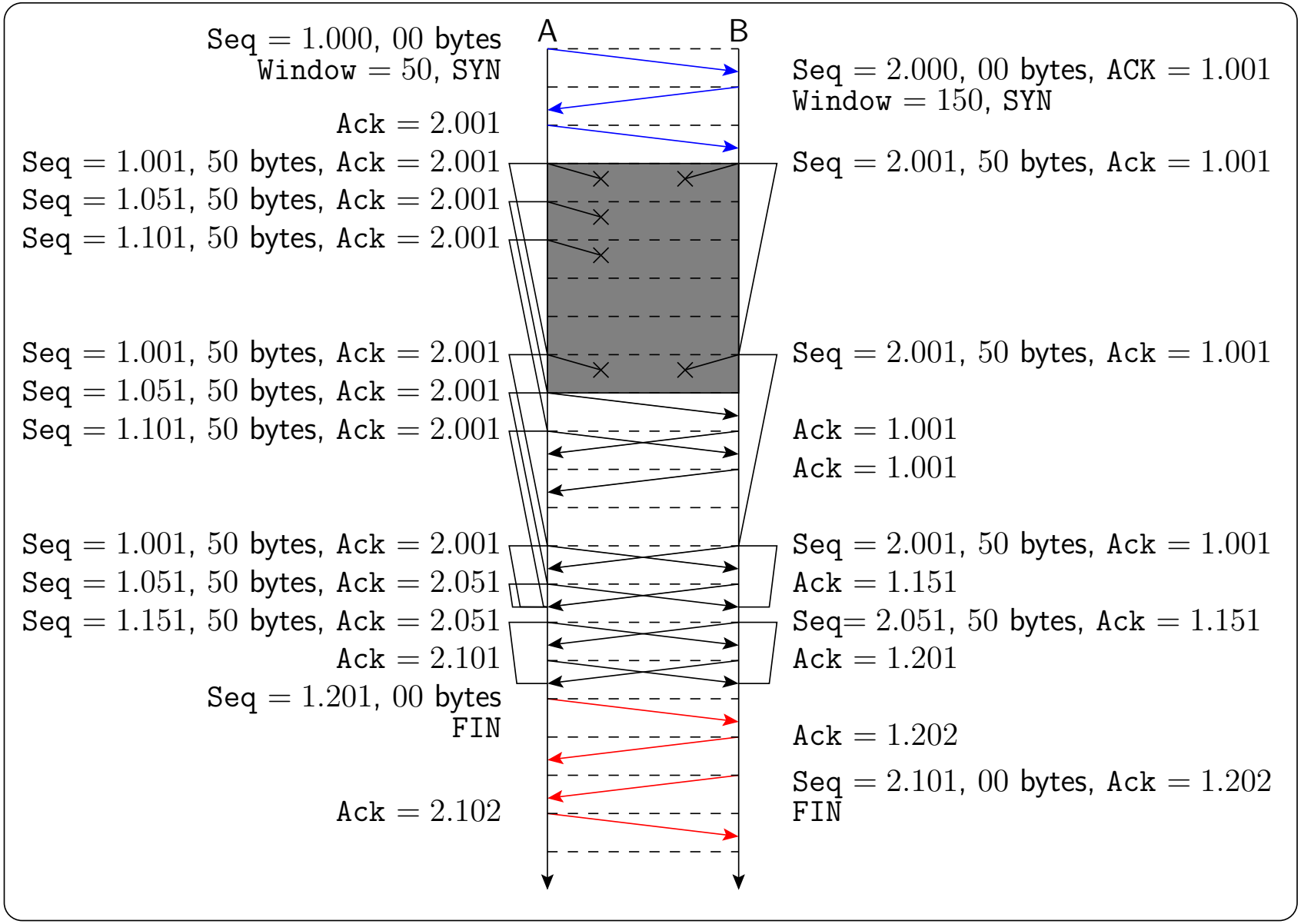
- $0 \leq \rho \leq 1$ es otro factor que hace lo mismo con la desviación estadística del RTT.
- Un ejemplo real [14]:

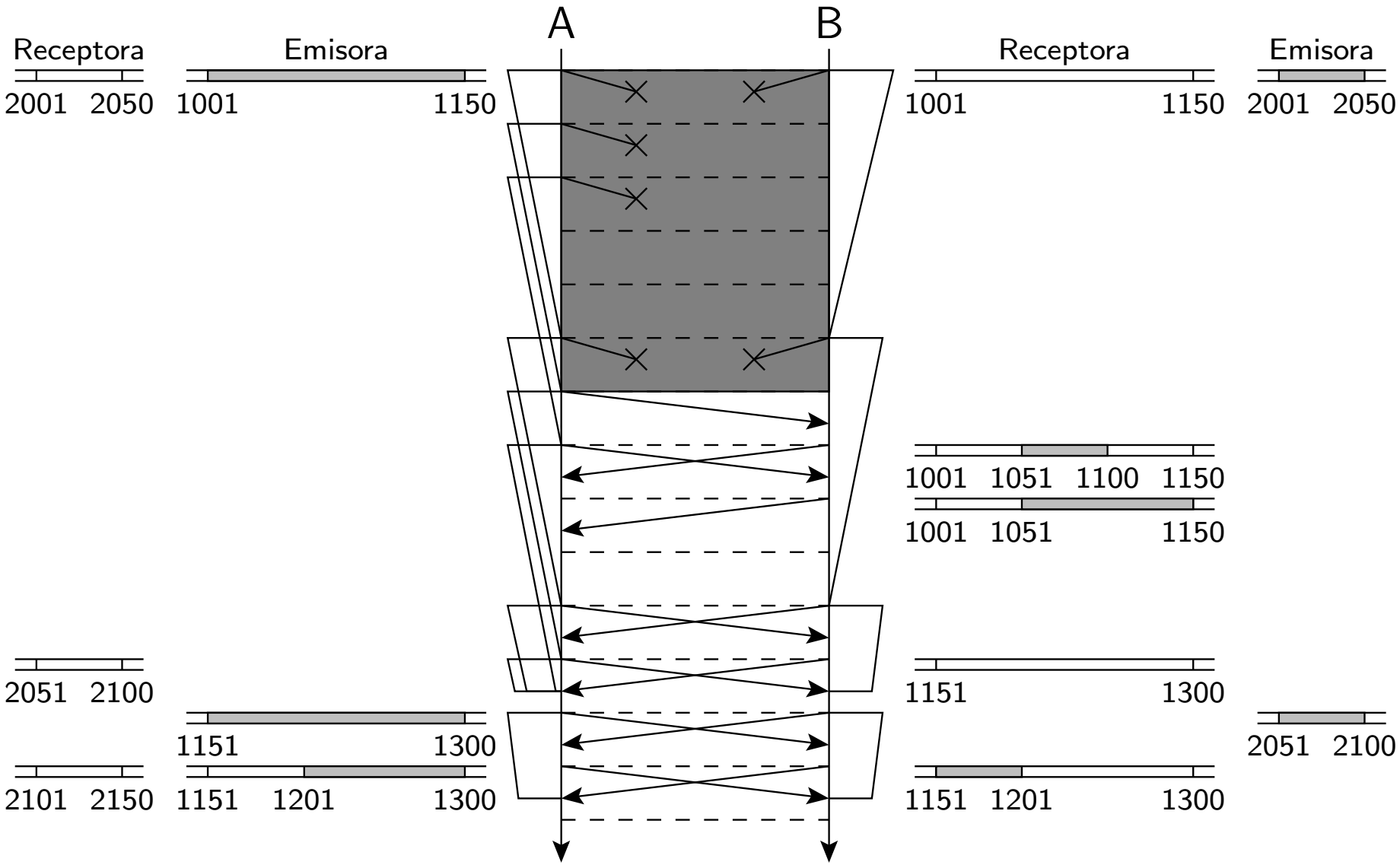


15.6. Ejemplos

Enunciado: *El host A desea enviar al host B* 200 *bytes de datos y el host B al host A* 100 *bytes de datos, utilizando el TCP. Por simplicidad, suponer que las transmisiones sólo se producen al comienzo de unos tics de reloj de periodo constante y que ningún segmento tarda más de un tic en llegar al otro extremo. Suponer además que el MSS* $= 50$ *bytes para ambas estaciones, que el tamaño de las ventanas receptoras queda determinado durante el establecimiento de la conexión (*50 *bytes para A y* 150 *bytes para B), que los segmentos enviados entre el 4º y 9º tic de reloj (es decir, justo después del establecimiento de la conexión) son corrompidos por el ruido, y que nunca se retrasan los ACK's. Dibujar el time-line asociado suponiendo que el número de secuencia inicial utilizado por A es* 1.000 *y por B es* 2.000*. Finalmente supóngase un TimeOut de* 5 *tics.*

Solución:


A
B
Seq = 1.000, 00 bytes
Window = 50, SYN
Seq = 2.000, 00 bytes, ACK = 1.001
Window = 150, SYN
Ack = 2.001
Seq = 1.001, 50 bytes, Ack = 2.001
Seq = 2.001, 50 bytes, Ack = 1.001
Seq = 1.051, 50 bytes, Ack = 2.001
Seq = 1.101, 50 bytes, Ack = 2.001
Seq = 1.001, 50 bytes, Ack = 2.001
Seq = 2.001, 50 bytes, Ack = 1.001
Seq = 1.051, 50 bytes, Ack = 2.001
Seq = 1.101, 50 bytes, Ack = 2.001
Ack = 1.001
Ack = 1.001
Seq = 1.001, 50 bytes, Ack = 2.001
Seq = 2.001, 50 bytes, Ack = 1.001
Seq = 1.051, 50 bytes, Ack = 2.051
Ack = 1.151
Seq = 1.151, 50 bytes, Ack = 2.051
Seq= 2.051, 50 bytes, Ack = 1.151
Ack = 2.101
Ack = 1.201
Seq = 1.201, 00 bytes
FIN
Ack = 1.202
Seq = 2.101, 00 bytes, Ack = 1.202
FIN
Ack = 2.102


A
B
Receptora
2001 2050
Emisora
1001 1150
Receptora
1001 1150
Emisora
2001 2050
1001 1051 1100 1150
1001 1051 1150
2051 2100
1151 1300
1151 1300
2051 2100
2101 2150
1151 1201 1300
1151 1201 1300

Enunciado: *Estamos transmitiendo ficheros entre dos hosts mediante TCP y observamos que la tasa de transmisión real es muy baja comparada con la capacidad física del enlace. Para mitigar este problema se proponen las siguientes soluciones (comentar su efectividad).*

1. *Se ha observado que la latencia entre los nodos es muy alta por lo que se adopta aumentar el tamaño de las ventanas emisoras en ambos nodos.*

   Solución: Para maximizar la tasa de transmisión el tamaño de las ventanas emisoras debe ser siempre mayor o igual que el producto RTT×capacidad del enlace. Por lo tanto, aumentar el tamaño de las ventanas puede ayudar.

2. *El porcentaje de segmentos perdidos es muy alto, aunque la latencia es muy baja. En este caso se adopta disminuir el timeout en ambos nodos.*

   Solución: El mecanismo que el TCP posee para solucionar los errores de transmisión es la retransmisión de las tramas tras expirar el correspondiente temporizador, lo que significa que

si la latencia es muy baja, entonces el time-out también debería serlo. Por lo tanto, disminuir los tiempos de reenvío puede ayudar.

3. *Se observa que el número de segmentos reenviados innecesariamente es muy alto. En este caso se propone aumentar el tamaño de las ventanas emisoras en ambos nodos.*

   Solución: El tamaño de las ventanas no afecta al instante en que los segmentos son reenviados. Por lo tanto, esta medida no mejorará las tasas de transmisión. Lo que ayudaría es aumentar los time-outs.

4. *Se observa que durante largos periodos de tiempo cesa la transmisión de datos por llenarse la ventana emisora del emisor. Como solución se propone disminuir el tamaño de la ventana emisora del emisor, pero aumentando su time-out.*

   Solución: Si la ventana emisora en el emisor permanece largos periodos de tiempo llena es porque la red no es suficientemente rápida o porque el receptor es demasiado lento, si los comparamos con el emisor. Disminuir el tamaño de

la ventana emisora en el emisor nunca mejorará la tasa de transmisión, independientemente de cómo sea el time-out. Sin embargo, aumentar el time-out podría ayudar ya que una red lenta y/o un receptor lento implican mayores RTT's con lo que el time-out debería ser grande.

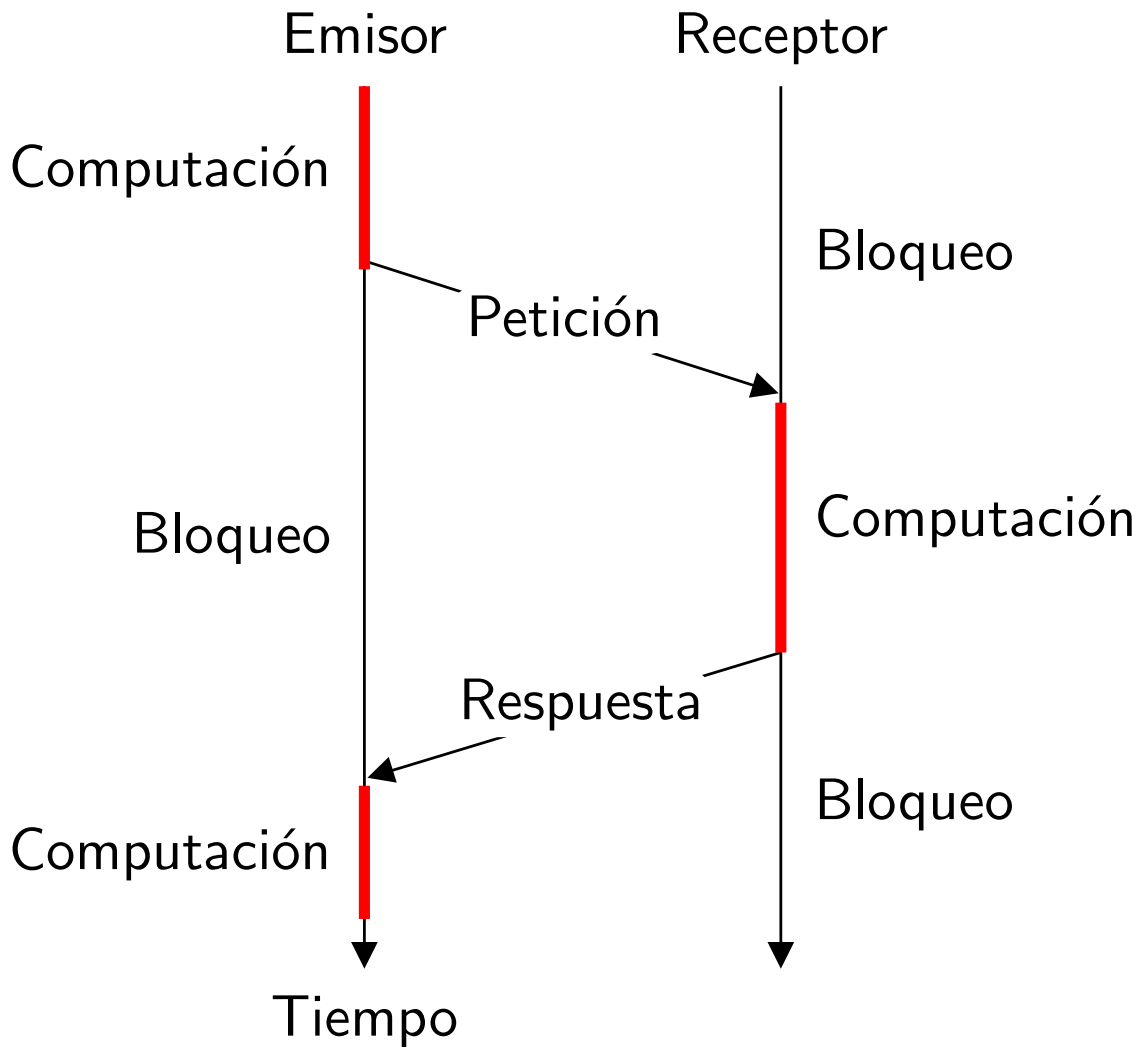
Capítulo 16

El mecanismo RPC (Remote Procedure Call)

16.1. Características

- Basado en el paradigma de comunicación cliente/servidor.
- Permite que el cliente invoque procedimientos (código ejecutable perteneciente a un determinado proceso) en el servidor [22].
- La sintaxis para el cliente es la misma que si invocara a un procedimiento local (de ahí su nombre).

- Al igual que ocurre con la ejecución de un procedimiento local, el proceso llamador espera (bloqueado) a que el procedimiento remoto termine de ejecutarse.

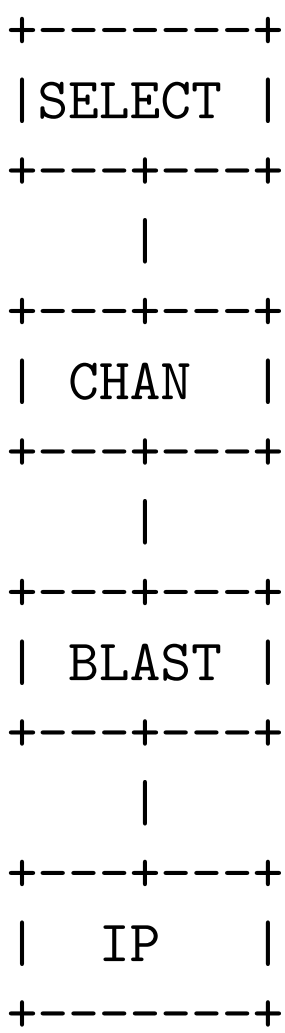


- Para que las aplicaciones usen el RPC, éste debe estar soportado por el compilador.
- Utilizado por aplicaciones como el NFS.

16.2. Microprotocolos del mecanismo RPC

- RPC utiliza tres microprotocolos:
  1. BLAST: fragmenta y ensambla mensajes grandes.
  2. CHAN: sincroniza los mensajes de petición y respuesta.
  3. SELECT: encamina los mensajes de petición al proceso correcto.

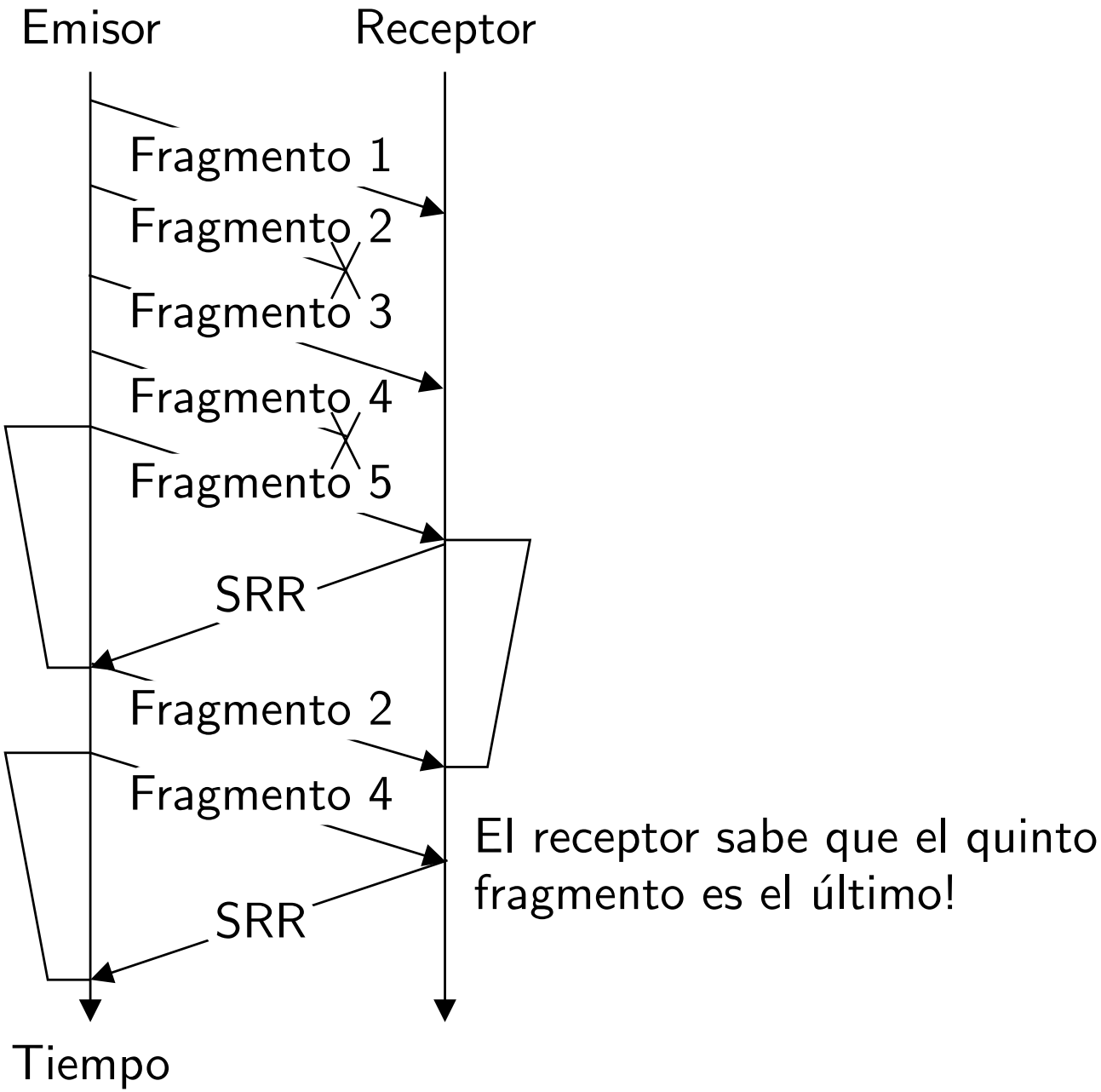
- Con todo esto, la pila de protocolos más simple que puede usar RPC sería:



16.2.1. BLAST

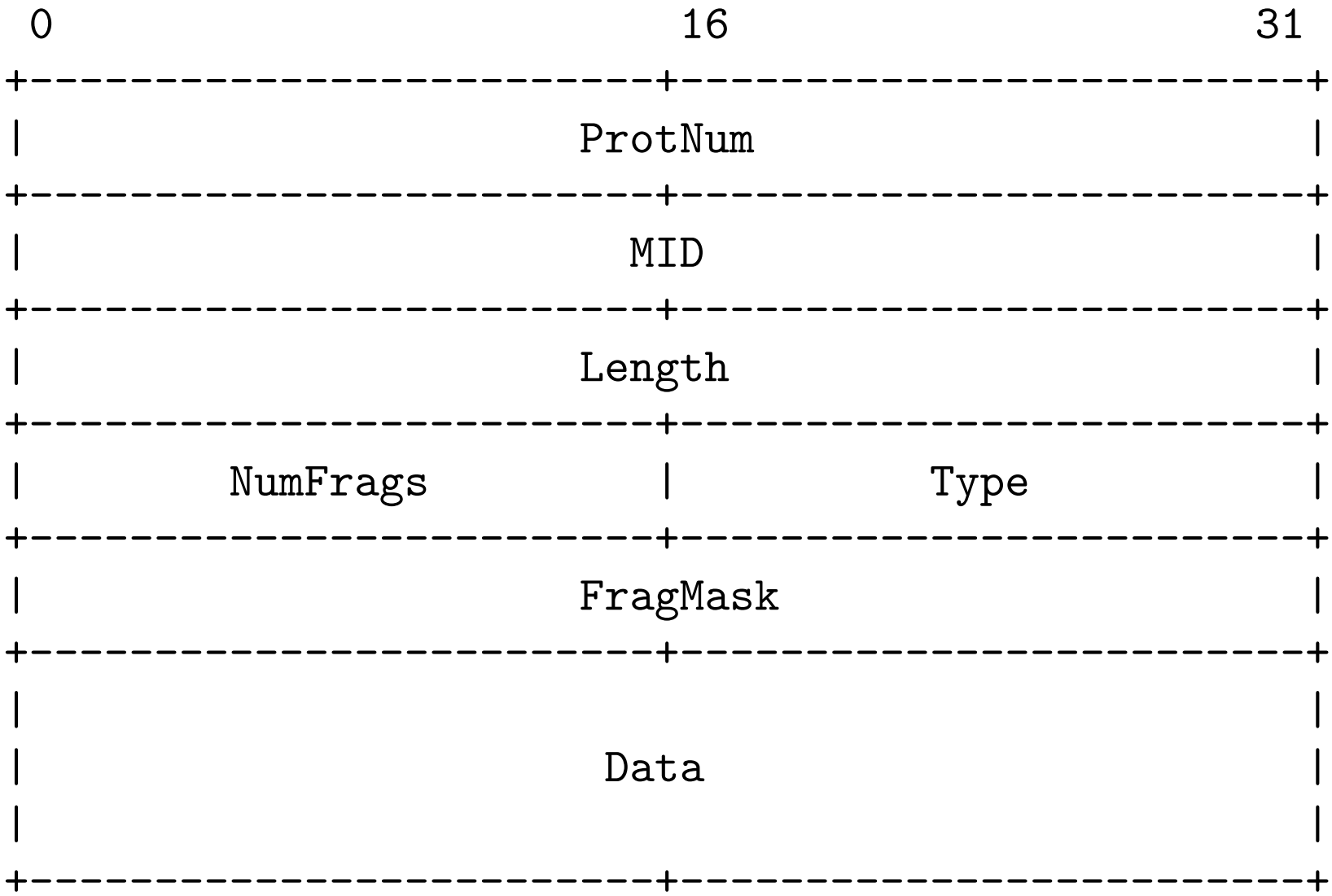
- Acomoda la longitud de los mensajes al MTU (Maximun Transfer Unit) de la red.[1]
- Semejante al TCP, aunque no está orientado a la transmisión de un flujo de datos.
- Soluciona la pérdida de paquetes mediante retransmisión selectiva (SRR = Selective Retransmission Request). Ejemplo:

[1] El IP también tiene esta funcionalidad, pero si un fragmento se pierde BLAST lo retransmite seletivamente. El IP retransmitiría todo el mensaje.


Emisor
Receptor
Fragmento 1
Fragmento 2
Fragmento 3
Fragmento 4
Fragmento 5
SRR
Fragmento 2
Fragmento 4
El receptor sabe que el quinto fragmento es el último!
SRR
Tiempo

- BLAST no garantiza la transmisión sin errores. Por ejemplo, si se pierden todos los paquetes de datos, no se enviará ningún SRR (que sólo se envían cuando hay errores). Por tanto, el mensaje se perderá.

- Cabecera de un paquete BLAST:



`ProtNum`: protocolo que utiliza BLAST.

`MID`: número de secuencia que identifica de forma única el mensaje.

`Length`: longitud del fragmento.

`NumFrags`: número de fragmentos del mensaje.

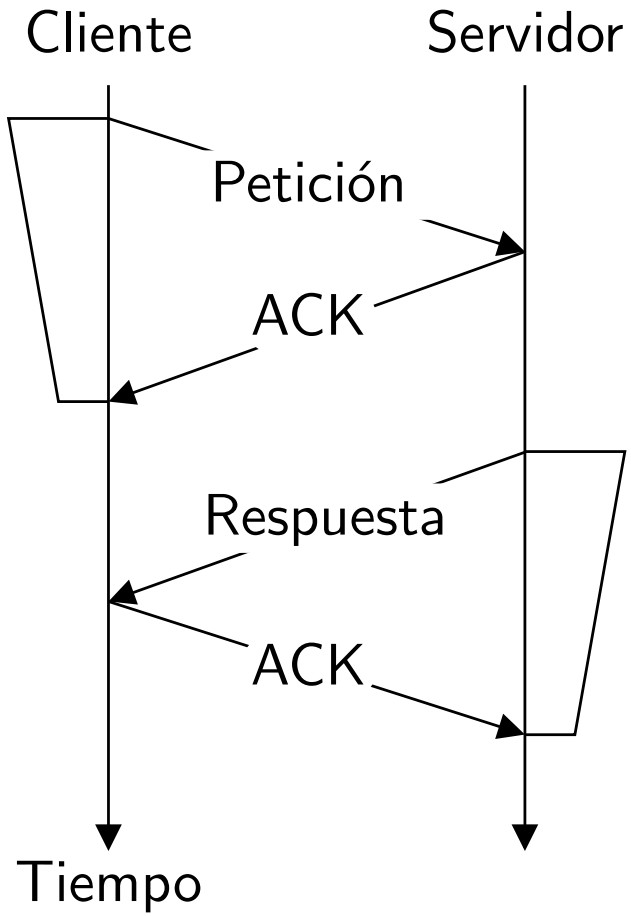
`Type`: DATA | SRR.

`FragMask`: Cuando se trata de un paquete con datos, sólo un bit puede estar a 1, indicando el índice del paquete dentro de la secuencia de fragmentación. Cuando se trata de un paquete SRR, un bit a 1 indica qué paquete de datos debe ser retransmitido. Nótese que no se pueden transmitir mensajes que al fragmentarse generan más de 32 paquetes.

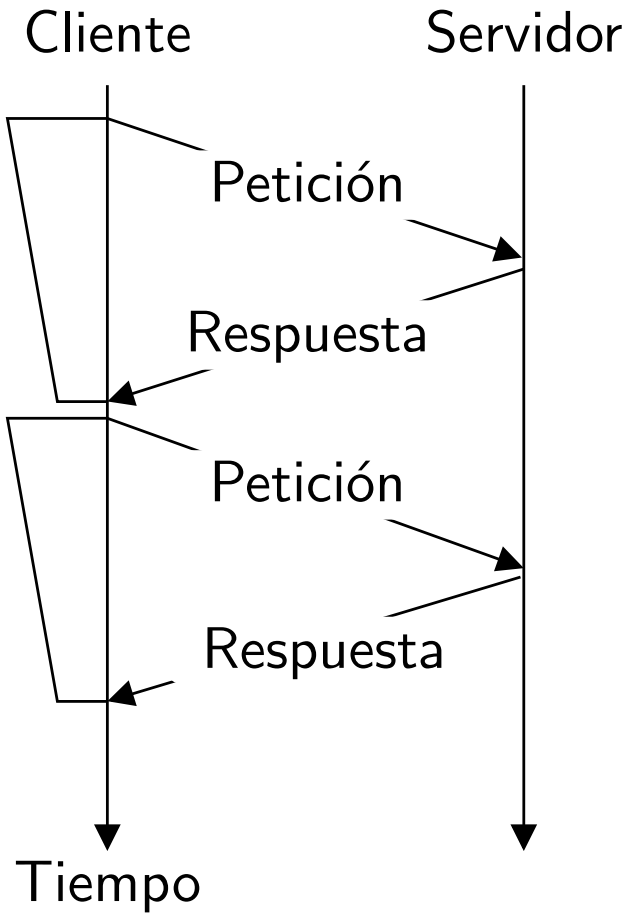
16.2.2. CHAN(nel)

- Implementa el mecanismo petición/respuesta del RPC.
- Convierte al RPC en fiable.
- Sincroniza a los procesos que se comunican[2] usando un protocolo en el que cada mensaje se reconoce positivamente (ACK):

[2] Sincronización = la operación `send()` termina cuando el receptor ha recibido los datos correctamente y además, el emisor lo sabe y recibe alguna contestación. Se dice en este caso que el `send()` es bloqueante.



Funcionamiento básico

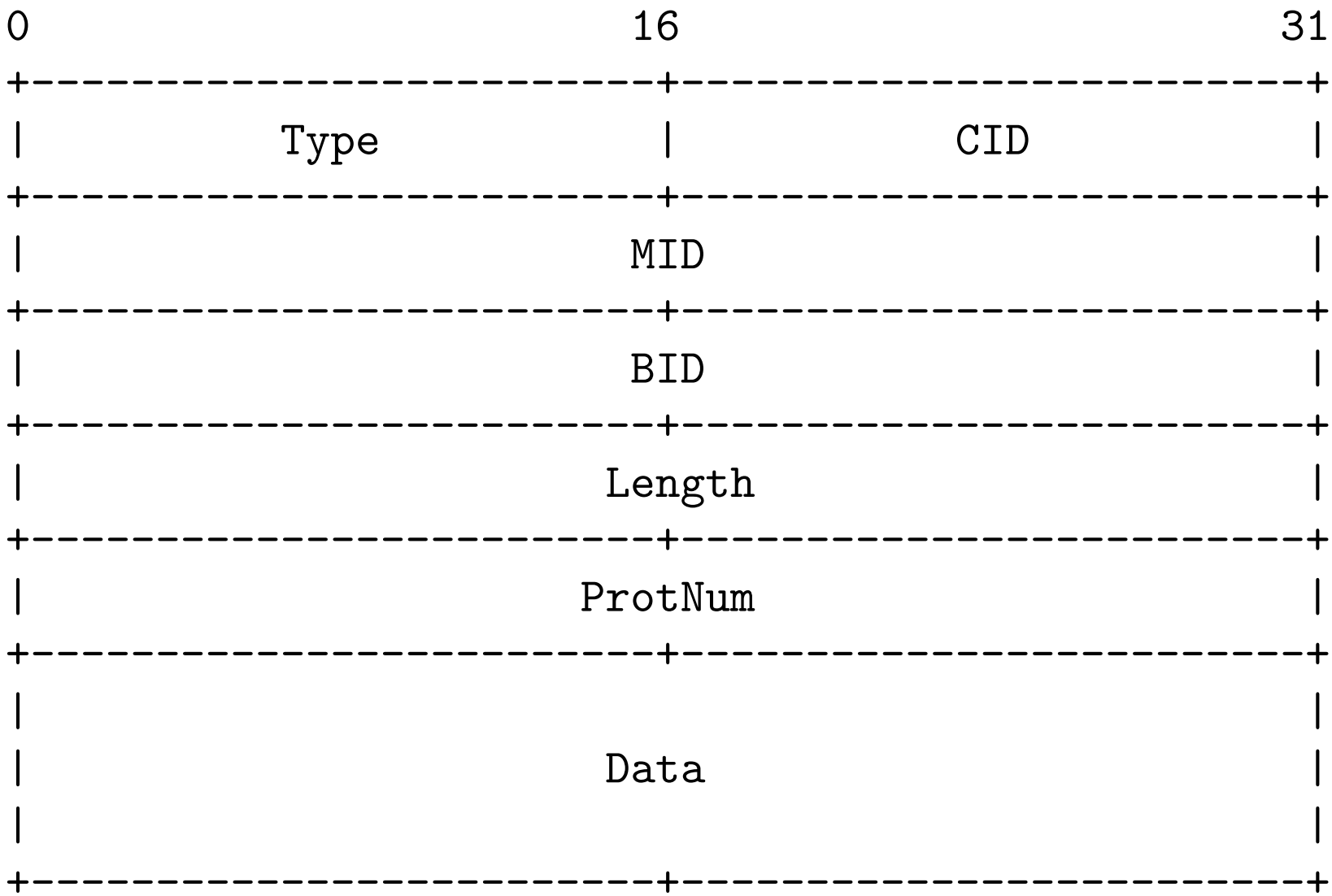


Uso de ACKs implícitos

aunque existe un modo donde se suprimen los reconocimientos para acelerar la comunicación.

- Se crea un canal por cada interacción petición/respuesta (llamada a un procedimiento remoto) que se produce.
- Más de un canal puede usarse al mismo tiempo. Esto es muy útil si las peticiones tardan mucho tiempo en responderse (pipelining).

- Formato de la cabecera CHAN:



Type: REQ(uest) | REP(ly) | ACK(nowlegment) | PROBE (are you

alive?).

`CID (Channel ID):` número de secuencia que identifica de forma única el canal (2^{16} canales simultaneos).

`MID (Message ID):` número de secuencia que identifica de forma única el mensaje. Sirve para descartar duplicados.

`BID (Boot ID):` número de secuencia que identifica de forma única el número de rebotes de la máquina. Sirve, junto con el MID, para descartar duplicados.

`Length:` longitud del mensaje.

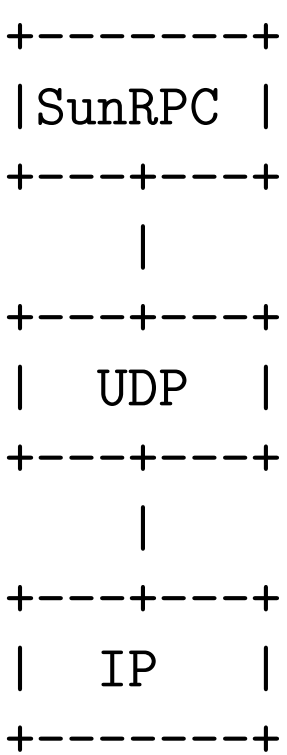
`ProtNum:` protocolo que usa CHAN.

16.2.3. SELECT

- Implementa la funcionalidad del desmultiplexado dentro del RPC: En el servidor hay un número determinado de procedimientos que pueden ser llamados por el cliente. Dichos procedimientos se enumeran y SELECT permite que el cliente pueda invocar a uno en concreto.
- La forma de enumeración más frecuente es la jerárquica: Cada proceso en el servidor tiene un número de proceso, y cada proceso enumera internamente sus procedimientos. El número de bits dedicados a la selección depende de la versión del RPC y del SO.

16.3. El caso particular de SunRPC

- SunRPC es una versión del sistema RPC desarrollado por Sun Microsystems para su SunOS. Es la implementación del RPC más extendida.
- Implementa una versión del RPC distinta de la planteada anteriormente, cuyo grafo de protocolos consiste en:



- El IP sustituye a BLAST (aunque con cierta pérdida de eficiencia) en la tarea de fragmentar los mensajes demasiado largos.
- El UDP sustituye en parte a SELECT ya que permite desmultiplexar al proceso correcto usando un puerto.
- Finalmente, SunRPC implementa la funcionalidad de CHAN y de SELECT, aunque de una forma diferente:
  - SunRPC corre como un demonio llamado *Port Mapper* que escucha, por defecto, en el puerto 111.
  - Cuando un cliente remoto solicita la ejecución de un procedimiento local, primero debe conocer en qué puerto está escuchando el proceso local que posee el procedimiento. Esta información la obtiene preguntando al *Port Mapper*.
  - Seguidamente, el cliente realiza la petición RPC al correspondiente proceso utilizando el puerto retornado en la anterior consulta.

- Los clientes normalmente cachean el resultado de la consulta realizada al *Port Mapper*, lo que no degrada el rendimiento en sucesiva llamadas a procedimientos remotos.

16.4. Otras implementaciones

- Las versiones actuales de RPC tienden a usar el TCP en lugar del UDP por diversas razones:

  1. En muchos cortafuegos el tráfico UDP está restringido.
  2. El mecanismo de control de la congestión del TCP es necesario en las transmisiones a larga distancia.
  3. El TCP es fiable mientras que el UDP no lo es.
  4. La pérdida de un segmento no implica la retransmisión del mensaje completo.

Parte IV

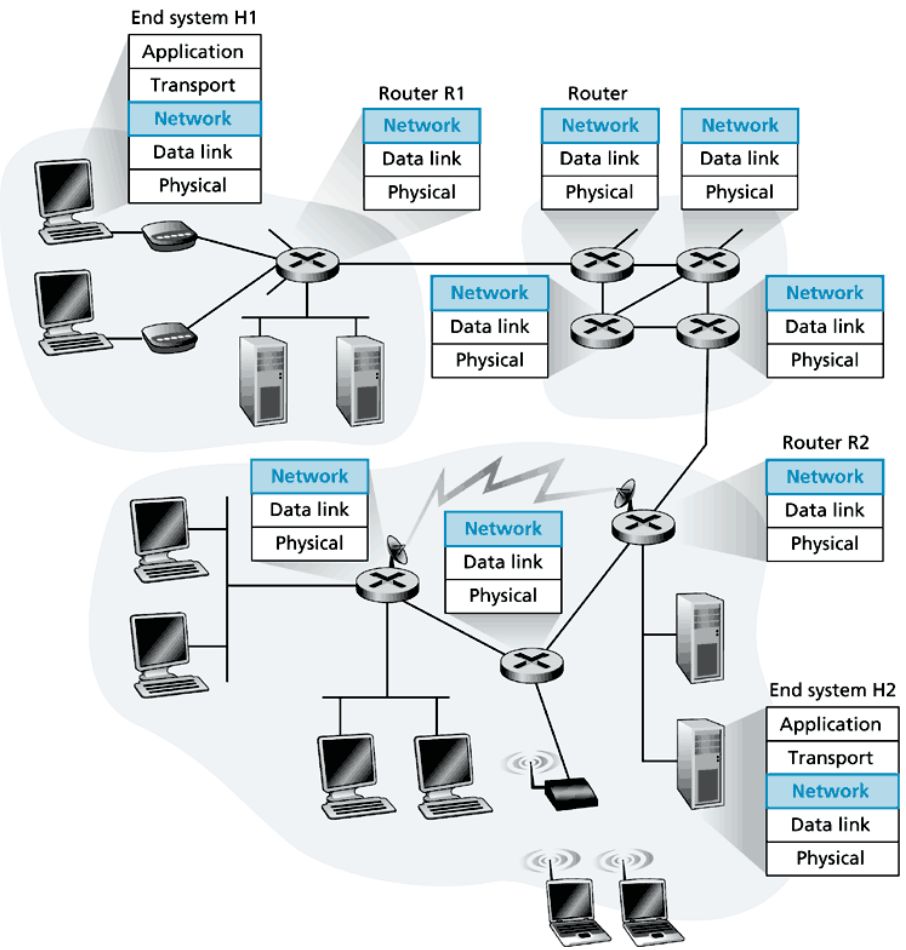
La capa de red

Capítulo 17

Servicios de la capa de red

17.1. El modelo de servicio

- Comunicación de paquetes (datagramas) entre hosts.
- La capa de red (en concreto el IP) corre en todos los routers y hosts de Internet.



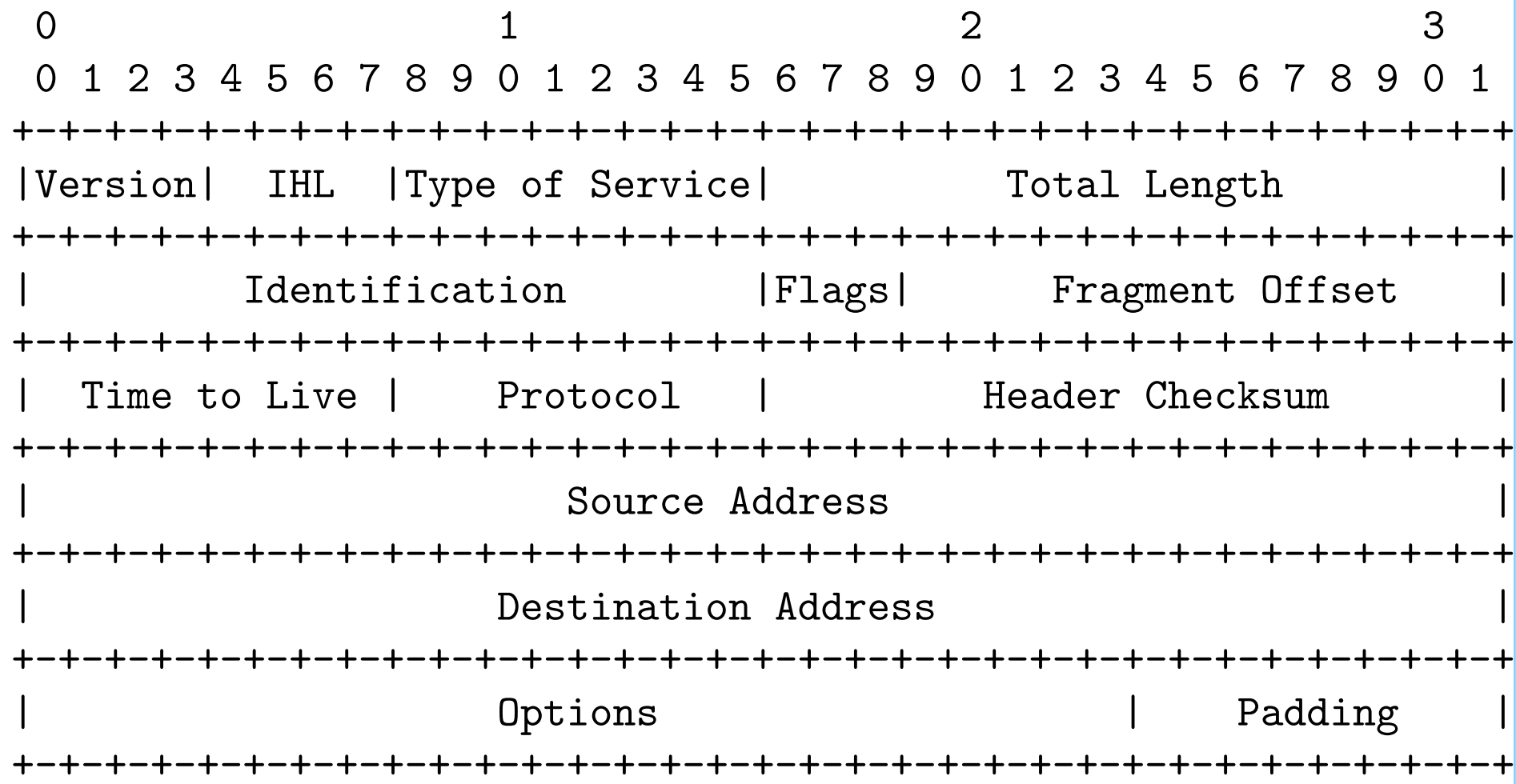
- Es la responsable del encaminamiento (forwarding) de los datagramas y del routing (actualización de las tablas encaminamiento) [24, 4, 14].
- No existen conexiones (este concepto sólo se maneja a nivel del TCP).
- Servicio mínimo:
  - No existe control de flujo, ni de errores, ni de congestión.
  - No se garantiza la tasa de transmisión, independientemente del intervalo de tiempo utilizado.

  ¿Realmente *best-effort*?
- Portabilidad extrema:
  - El servicio definido por el IP es tan pobre que casi cualquier tecnología de red inventada hasta la fecha es capaz de proporcionarlo.
  - Gracias a su sencillez, el IP mantiene a los routers simples.

17.2. El IP (Internet Protocol)

- Se encarga del encaminamiento de los datagramas en Internet.
- Existen actualmente dos versiones: la IPv4 (RFC 791) que es la que en estos momentos se utiliza y la IPv6 (RFC 2460, RFC 2373) que se planea utilizar en un futuro próximo.
- IPv6 es compatible hacia atrás con IPv4 (IPv6 es capaz de encaminar paquetes IPv4).

17.2.1. Formato de la cabecera en IPv4



- `Version`: Versión del protocolo IP (4).
- `IHL` (Internet Header Length): Tamaño de la cabecera del paquete IP en palabras de 32 bits.
- `Type of Service` (TOS): Indicación de la calidad de servicio que se espera recibir por parte de los routers (tráfico multimedia, información sobre la congestión de la red, etc.).
- `Total Length`: Tamaño del datagrama IP (cabecera y datos) en bytes. Tamaños típicos son (debido principalmente a los MTU's de las redes) 1.500 bytes (Ethernet) y 576 bytes (PPP).
- `Identification`: Etiqueta creada por el emisor del paquete y que se utiliza cuando éste se fragmenta.
- `Flags`:

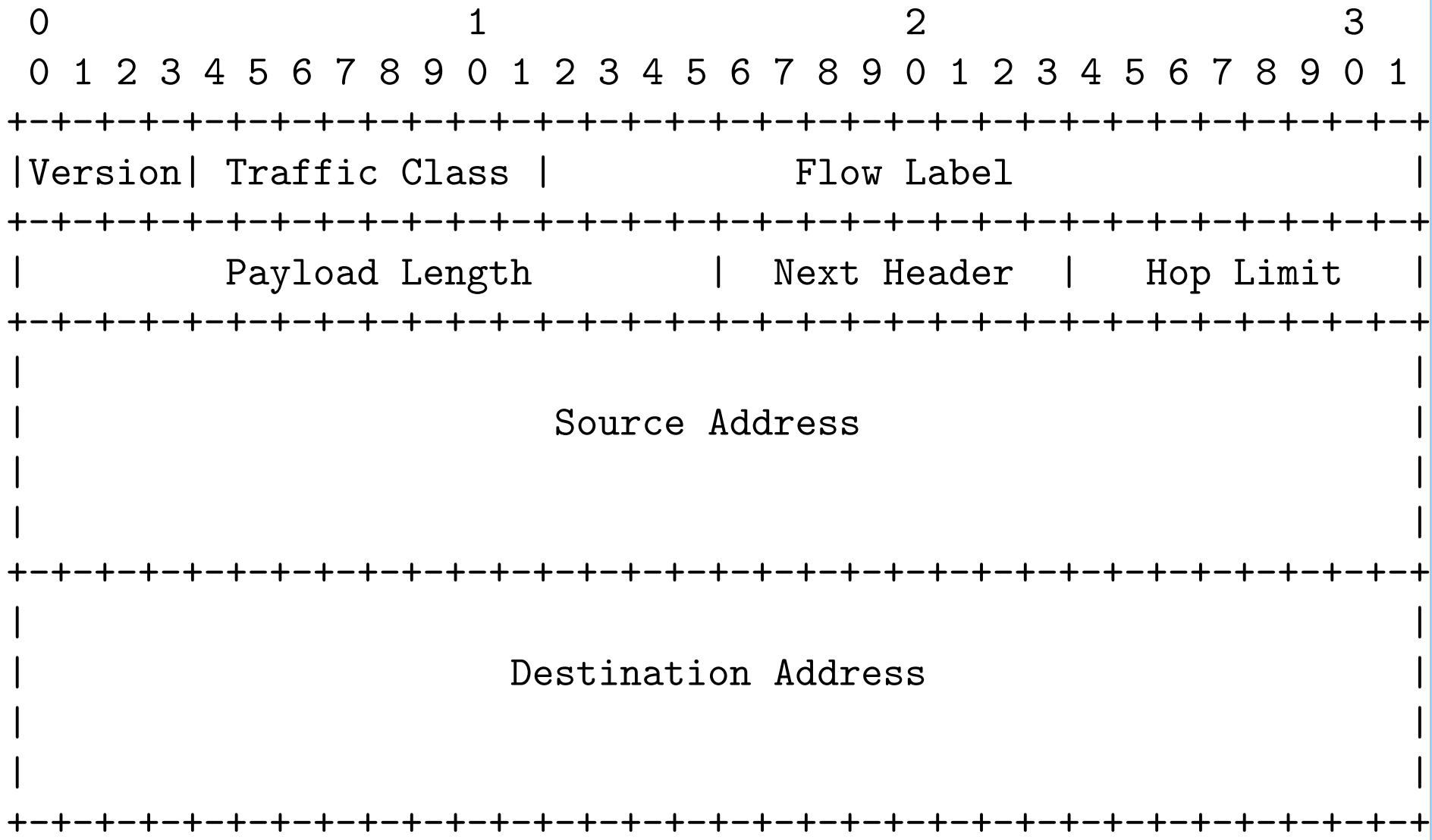
```
Bit 0: reserved, must be zero.
Bit 1: (DF) 0 = May Fragment,  1 = Don't Fragment.
Bit 2: (MF) 0 = Last Fragment, 1 = More Fragments.
```

- `Fragment Offset`: Offset del paquete cuando se ha fragmentado en palabras de 8 bytes.
- `Time to Live` (TTL): Valor que se decrementa cada vez que el paquete es retransmitido por un router. Cuando TTL $= 0$ entonces el paquete se destruye[1].
- `Protocol`: Protocolo al que va dirigido el paquete (RFC's 790, 1700 y 3232). Por ejemplo, cuando se transporta un paquete UDP se utiliza el 17, para TCP el 6, para ICMP el 1, etc.
- `Header Checksum`: Código de detección de errores que sirve para desechar el paquete si se han producido errores de transmisión en la cabecera. Es una suma de todas las palabras de 16 bits de (sólo) la cabecera IP usando aritmética en complemento a 1 (RFC 1071). Este valor es recalculado en cada hop (salto) porque en cada uno de ellos el TTL se decrementa.
- `Source Address`: Dir IP del host que generó el paquete.

[1] Y se genera un paquete ICMP para el emisor.

- `Destination Address`: Dir IP del host al que va dirigido el paquete.
- `Options`: Campo de longitud variable (desde 0 bytes) que se utiliza para diferentes propósitos (almacenar rutas, colocar estampas de tiempo, etc.).
- `Padding`: Bits a 0 rellenando la cabecera hasta tener un tamaño múltiplo de 32 bits.

17.2.2. Formato de la cabecera en IPv6



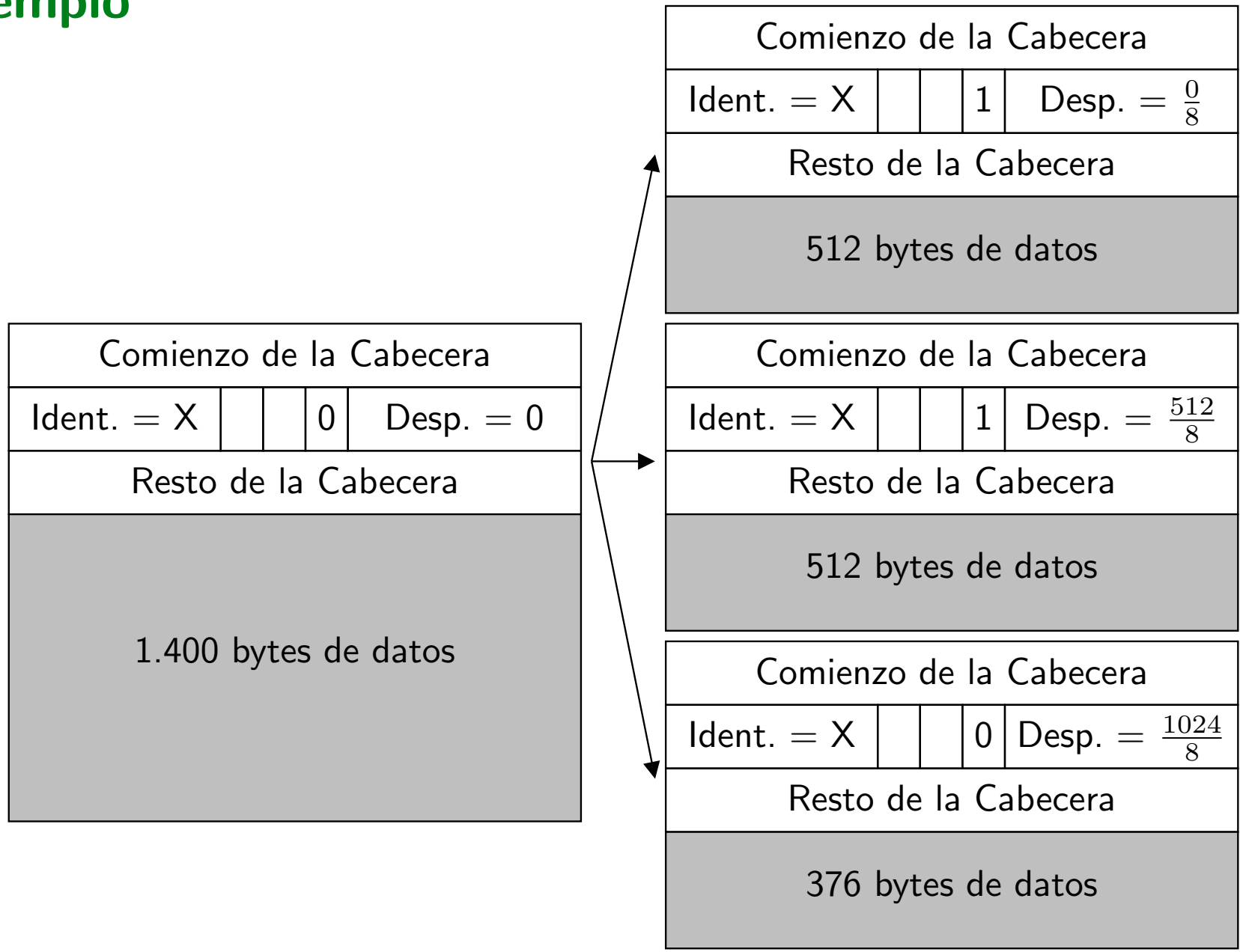
- `Version`: Versión del protocolo (6).
- `Traffic Class`: Identifica el tipo de tráfico. Por ejemplo, sirve para diferenciar el tráfico multimedia del que no es sensible al tiempo.
- `Flow Label`: Identifica a los paquetes de un mismo flujo de datos (tráfico multimedia).
- `Payload Length`: Número de bytes de datos transportados (sin considerar la cabecera que es de tamaño fijo – 40 bytes –).
- `Next Header`: Identifica el protocolo que utiliza el IP (igual que IPv4).
- `Hop Limit`: TTL.
- `Source Address`: Dir IP del host origen del paquete.
- `Destination Address`: Dir IP del host destino del paquete.

17.2.3. Fragmentación y ensamblaje

- Cada tecnología de red posee un límite en el tamaño máximo del paquete a transmitir (MTU). Por ejemplo, en Ethernet el límite es de 1.500 bytes.
- El IP (en los hosts y en los routers), antes de enviar los datagramas los fragmenta de forma que ninguno de ellos tiene un tamaño superior al MTU de la red directamente conectada.
- Durante el trayecto los datagramas pueden ser re-fragmentados si se atraviesa una red de menor MTU.
- Sólo el IP del host destino realiza el ensamblaje (nunca los routers). Si alguno de los fragmentos no llega correctamente se desechan el resto de fragmentos.
- Sólo en IPv4. No en IPv6[2].

[2] Si el MTU es demasiado pequeño, el paquete IPv6 se destruye y se genera un paquete ICMP para el emisor.

Ejemplo


Comienzo de la Cabecera
Ident. = X
0
Desp. = 0
Resto de la Cabecera
1.400 bytes de datos
Comienzo de la Cabecera
Ident. = X
1
Desp. = 0/8
Resto de la Cabecera
512 bytes de datos
Comienzo de la Cabecera
Ident. = X
1
Desp. = 512/8
Resto de la Cabecera
512 bytes de datos
Comienzo de la Cabecera
Ident. = X
0
Desp. = 1024/8
Resto de la Cabecera
376 bytes de datos

17.3. El ICMP (Internet Control Message Protocol)

- RFC 792.
- Informe de errores.
- Generado por routers y hosts.
- Los mensajes se encapsulan en paquetes IP.
- Los mensajes contienen los primeros 8 bytes del paquete IP que produjo el mensaje ICMP. Esto se hace para que el receptor del paquete ICMP conozca el paquete IP que produjo el error [15].

- Mensajes ICMP:

Tipo	Código	Fuente/Mensaje
0	0	Nodo/Respuesta a un eco (ping)
3	0	Router/Red destino inalcanzable (no funciona)
3	1	Router/Host destino inalcanzable
3	2	Host/Protocolo destino inalcanzable
3	3	Host/Puerto destino inalcanzable
3	6	Router/Red destino desconocida (no existe)
3	7	Router/Host destino desconocido
4	0	Router/Apaciguar host (control de congestión)
8	0	Host/Petición de eco
9	0	Router/Anuncio de router
10	0	Router/Descubrimiento de router
11	0	Router/TTL expirado
12	0	Nodo/Cabecera IP errónea

Ejemplo: `traceroute`

- Programa de diagnóstico que permite conocer la ruta (routers) por los que pasan los paquetes que viajan desde nuestra máquina (la que ejecuta el programa `traceroute`) hasta la máquina destino.
- Uso:

```
traceroute [flags] nodo_destino
```

- Funciona aprovechando los servicios proporcionados por el ICMP (Internet Control Message Protocol), que se utiliza para conocer qué está ocurriendo en la red.
- Uno de los mensajes que el ICMP proporciona es el de tiempo excedido (time exceeded). Estos mensajes son generados sólo por los routers cuando no retransmiten un paquete porque su TTL (Time To Live) es 0.

- El campo TTL figura en todas las cabeceras de los paquetes que son transmitidos a través de Internet y sirven para controlar cómo de lejos van a viajar. Cada vez que un paquete atraviesa un router su TTL se decrementa.

- `traceroute` fuerza a que los distintos routers contesten con un ICMP TIME_EXCEEDED, enviando paquetes con un TTL que se va incrementando en 1 en cada iteración. Entonces `traceroute` calcula los tiempos de ida y vuelta o RTT (Round-Trip Time) en 3 ocasiones.

- Otro de los mensajes que el ICMP proporciona es el de puerto inalcanzable (unreacheable port). Este tipo de mensaje es generado únicamente por los hosts cuando les llega un paquete cuyo puerto destino no está siendo escuchado por ninguna aplicación.

- `traceroute` obtiene el RTT del host destino aprovechando el mensaje ICMP PORT_UNREACHEABLE generado por éste, ya que el puerto destino usado (el 33.434) no suele utilizarse para otro propósito.

- Un ejemplo de funcionamiento sería:

```
traceroute www.cica.es
traceroute to ataman.cica.es (150.214.4.16), 30 hops max, 38 byte packets
 1  rou118.ual.es (193.147.118.1)  0.701 ms  0.574 ms  0.480 ms
 2  11.0.0.5 (11.0.0.5)  0.840 ms  0.783 ms  0.767 ms
 3  192.168.1.1 (192.168.1.1)  0.653 ms  0.644 ms  0.608 ms
 4  almeria.cica.es (150.214.231.97)  1.795 ms  1.761 ms  2.740 ms
 5  jds-rt2-almeria.cica.es (150.214.0.33)  13.333 ms  13.354 ms  12.946 ms
 6  ataman.cica.es (150.214.4.16)  13.476 ms *  18.044 ms
```

- Para conocer más sobre esta utilidad, en una máquina Unix ejecutar `man traceroute` o visitar la página Web `www.traceroute.org`.

17.4. El DHCP (Dynamic Host Configuration Protocol)

- RFC 2131.
- Protocolo cliente-servidor, UDP, puerto 67.
- Los clientes lo utilizan para obtener de forma automática los parámetros de conexión (dir IP, máscara, gateway y DNS) de la red.
- Especialmente utilizado en el caso de hosts móviles.
- Típicamente se utiliza para asignar un pool de X dirs IP a Y hosts de forma dinámica, donde generalmente $X \leq Y$.
- Cuando un host arranca se pone en contacto con su servidor DHCP preguntando a la dir de broadcast[3] (255.255.255.255) y usando la dir IP origen 0.0.0.0.

[3] Lo que se envía a esta dir IP le llega a todos los nodos de la red local.

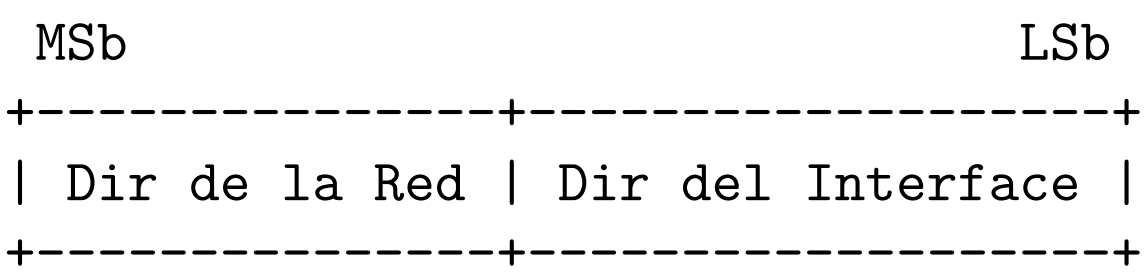
- En una misma red pueden existir varios servidores DHCP. En principio todos contestarían y es el cliente quien elige.
- Las configuraciones pueden asignarse de forma temporal o de forma indefinida.

Capítulo 18

Addressing (Direccionamiento)

18.1. Direccionamiento en IPv4

- Cada enlace que se conecta a un host o un router en Internet lo hace a través de un interface y cada interface tiene asociada una dir IP.[1]
- En IPv4 las dirs IP son de 32 bits y en IPv6 de 128 bits. Formato:



- Las dirs IP son jerárquicas. La primera parte identifica la red a la que está conectada el interface y la segunda parte, el interface dentro de la red.

[1] La ICANN (Internet Corporation for Assigned Names and Numbers, RFC 2050) es la autoridad global de asignación de dirs IP. Maneja también los servidores de nombres raíz. Los administradores de redes acceden a la ICANN a través de las respectivas autoridades regionales de registros de Internet (ejemplos: ARIN (American Registry for Internet Numbers), RIPE (Réseaux IP Européens) y APNIC (Asia Pacific Network Information Centre)).

- Por definición, *todos los interfaces de una misma red tienen la misma dirección de red*.
- En IPv4 se suele utilizar la "notación decimal punteada" para representar las dirs IP, donde cada byte se traduce a su forma decimal y se separa por un punto del resto de bytes de la dirección. Ejemplo:

$$00001111\ 00010001\ 00000001\ 00000010 = 15.17.1.2$$

- Para diferenciar en una dir IP la dirección de la red y la dirección del interface, se utiliza **la máscara de red**. En el caso de IPv4 son 32 bits y para IPv6 128. Las máscaras son siempre de la forma

$$1111...111000...000$$

donde se cumple que

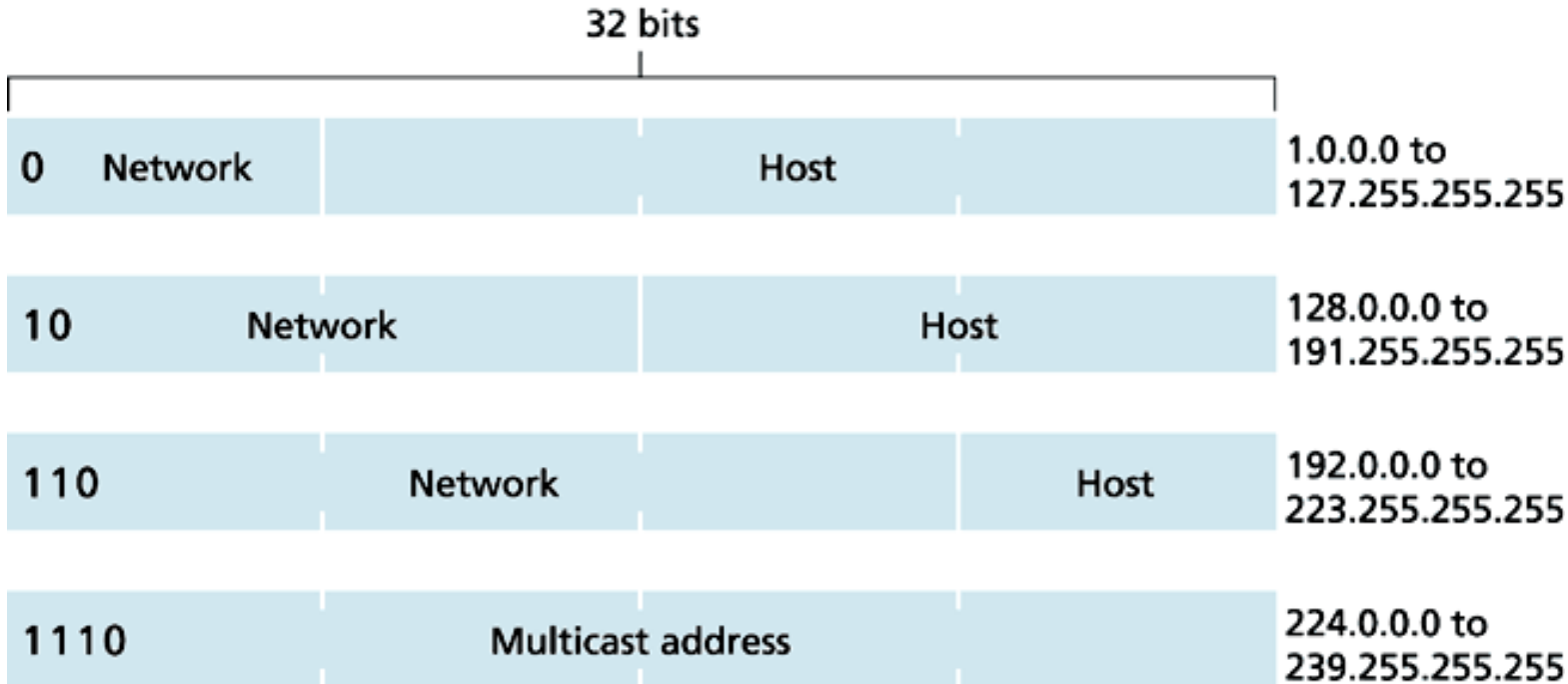
$$\text{dir\_IP AND máscara} = \text{dir\_Red}$$

y que

dir_IP MOD máscara = dir_Interface.

18.2. Clases de dirs IP

- Las redes IP se consideran de diferentes clases dependiendo de los bits más significativos [14]:
  - En IPv4:



- En IPv6:

Allocation	Prefix (binary)	Fraction of Address Space
Reserved	0000 0000	1/256
Unassigned	0000 0001	1/256
Reserved for NSAP (ATM) Allocation	0000 001	1/128
Reserved for IPX (Novell) Allocation	0000 010	1/128
Unassigned	0000 011	1/128
Unassigned	0000 1	1/32
Unassigned	0001	1/16
Aggregatable Global Unicast Addresses	001	1/8
Unassigned	010	1/8
Unassigned	011	1/8
Unassigned	100	1/8
Unassigned	101	1/8
Unassigned	110	1/8
Unassigned	1110	1/16
Unassigned	1111 0	1/32
Unassigned	1111 10	1/64
Unassigned	1111 110	1/128
Unassigned	1111 1110 0	1/512
Link-Local Unicast Addresses	1111 1110 10	1/1024
Site-Local Unicast Addresses	1111 1110 11	1/1024
Multicast Addresses	1111 1111	1/256

18.3. Sub-netting y dirs CIDR en IPv4

- Sub-netting en RFC 950.
- CIDR (Classless InterDomain Routing) en RFC 1519.
- Originalmente sólo se permitían tres clases de dirs IPv4 y por lo tanto sólo existían 3 tamaños posibles para las redes IP. El tipo de red se distinguía por sus primeros bits.

Clase	Bits Iniciales	Número de Redes	Número de Interfaces
A	0	2^7	2^{24}
B	10	2^{14}	2^{16}
C	110	2^{21}	2^8

- Los diseñadores de Internet pronto se dieron cuenta de que en muchos casos de producía un desperdicio considerable de dirs IP (por ejemplo, hacía falta una red de clase B para conectar a ella 257 interfaces).

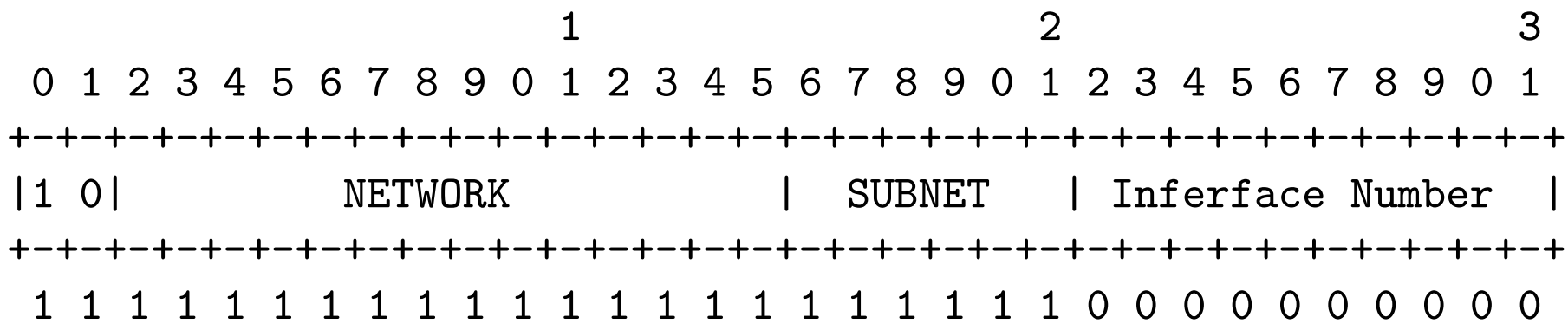
- La máscara de red (aparte de indicar la dirección de la red) define (independientemente de los bits iniciales de las dirs IP) el tamaño de la red. Así, una red de clase C se denota por

  X.Y.Z.0/24,

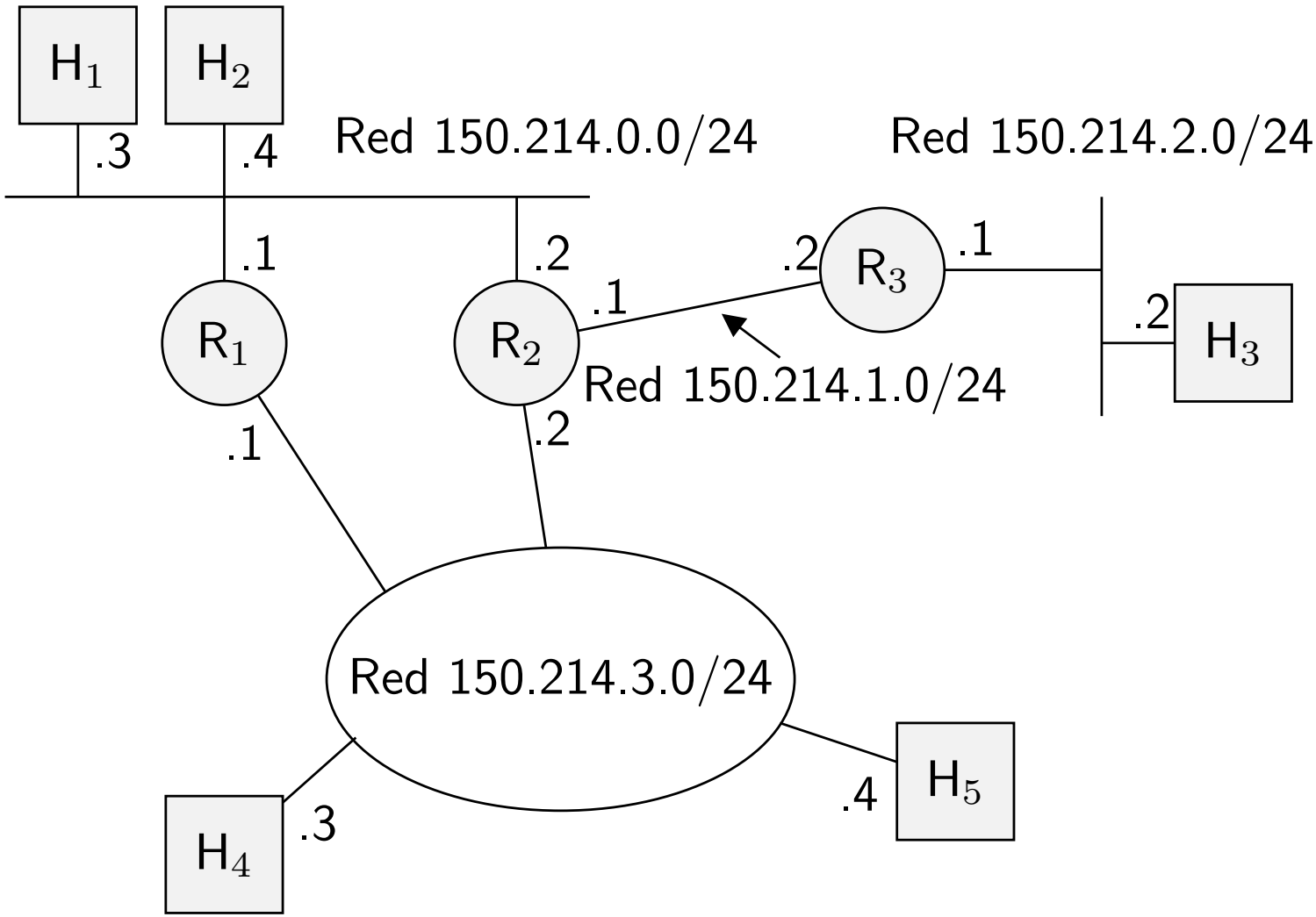
  porque la máscara de red, para todos los interfaces conectados a este tipo de red, posee 24 unos.

- A este tipo de dirs IP, donde la máscara de red puede tener cualquier número M de bits y por tanto la red puede contener hasta 2^{32-M} interfaces, se les llama dirs IP CIDR.

- Usando sub-netting, cualquier rango de dirs IP se puede dividir en conjuntos disjuntos de dirs IP y cada conjunto puede ser una (sub-)red diferente. Ejemplo (sub-netting en una red de clase B):



Ejemplo de red IP



18.4. Redes privadas

- RFC 1918.
- Permiten conectar más nodos a la red que el proporcionado por el espacio de dirs IPv4.
- Se implementan usando NAT.
- La IANA ha reservado los siguientes bloques de dirs IP privadas:

Bloque	Desde	Hasta
10.0.0.0/8	10.0.0.0	10.255.255.255
172.16.0.0/12	172.16.0.0	172.31.255.255
192.168.0.0/16	192.168.0.0	192.168.255.255

18.5. NAT (Network Address Translation)

- Se define en los RFC's 2663 y 3022.
- Se ejecuta en un router o un host que funciona como un router.
- El router tiene asignada sólo una dir IP pública. Esta pertenece al interface que conecta el router con Internet.
- Todos los interfaces de la red interna (normalmente privada) poseen dirs IP que no son visibles desde Internet.
- Todos los nodos de la red interna acceden a Internet mediante la IP del router NAT, en otras palabras, todos ellos utilizan la dir IP del interface público del router NAT.
- El router utiliza una tabla de traducción NAT para asociar conexiones

$$\text{[Dir IP Pública, puerto (router)]} \leftrightarrow \text{[Dir IP Privada, puerto (host)]}.$$

- La tabla NAT tiene 3 columnas:

Puerto público	Dir IP privada	Puerto privado

- Cuando un host privado envía un paquete para un host público, el router NAT inserta una entrada en la tabla de traducción NAT con un puerto libre y utiliza el puerto público asignado para todos los paquetes salientes con la misma

  (Dir IP Privada, Puerto privado).

- Cuando llega un paquete para una estación de la red privada, el router busca en su tabla de traducción NAT usando como clave

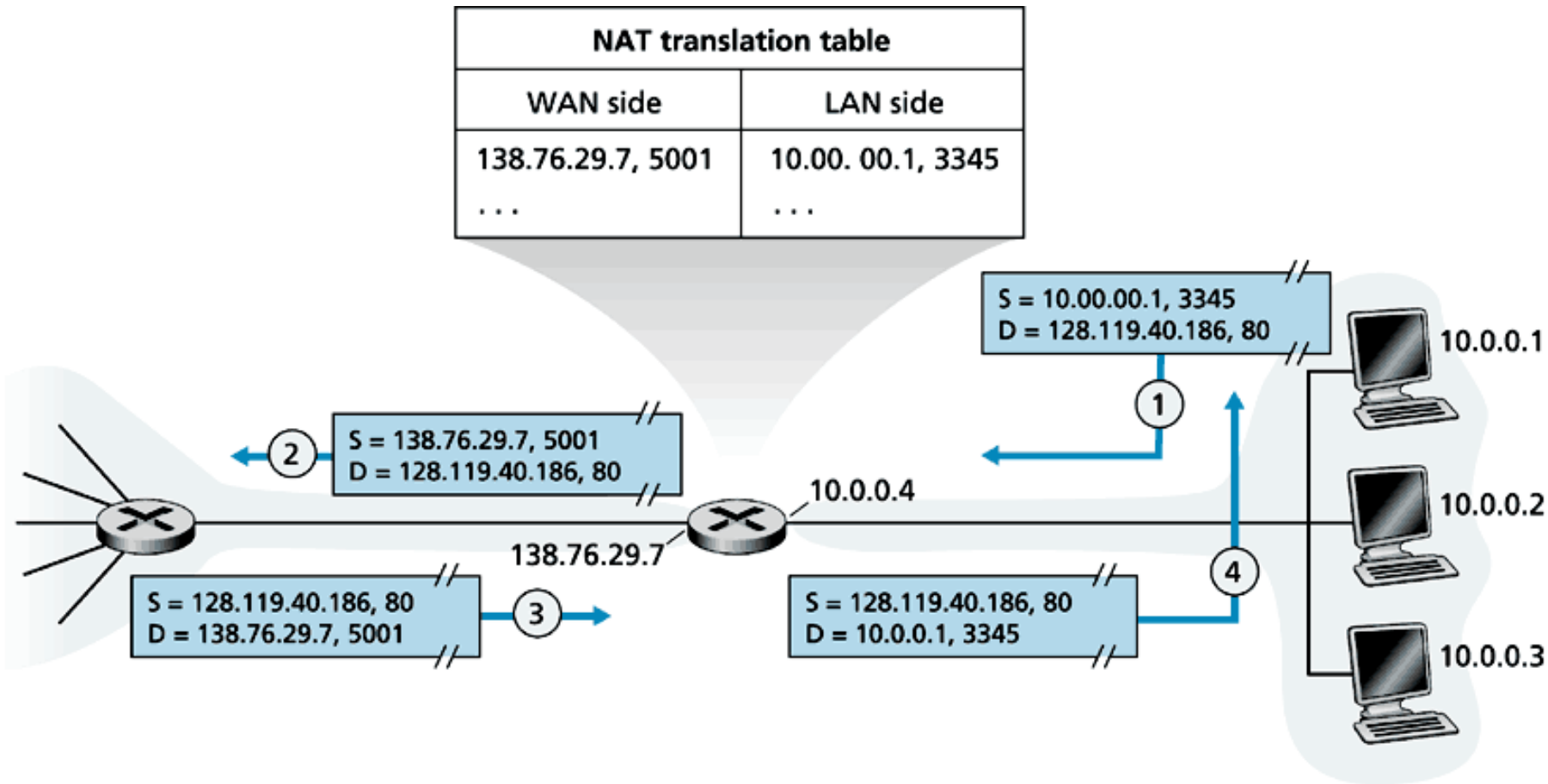
  (Puerto público)

  y encamina hacia la red privada el paquete usando la dir IP privada y el puerto (privado).

- Nótese que en cualquier caso no son posibles más de 2^{16} conexiones simultáneas a través del router NAT.

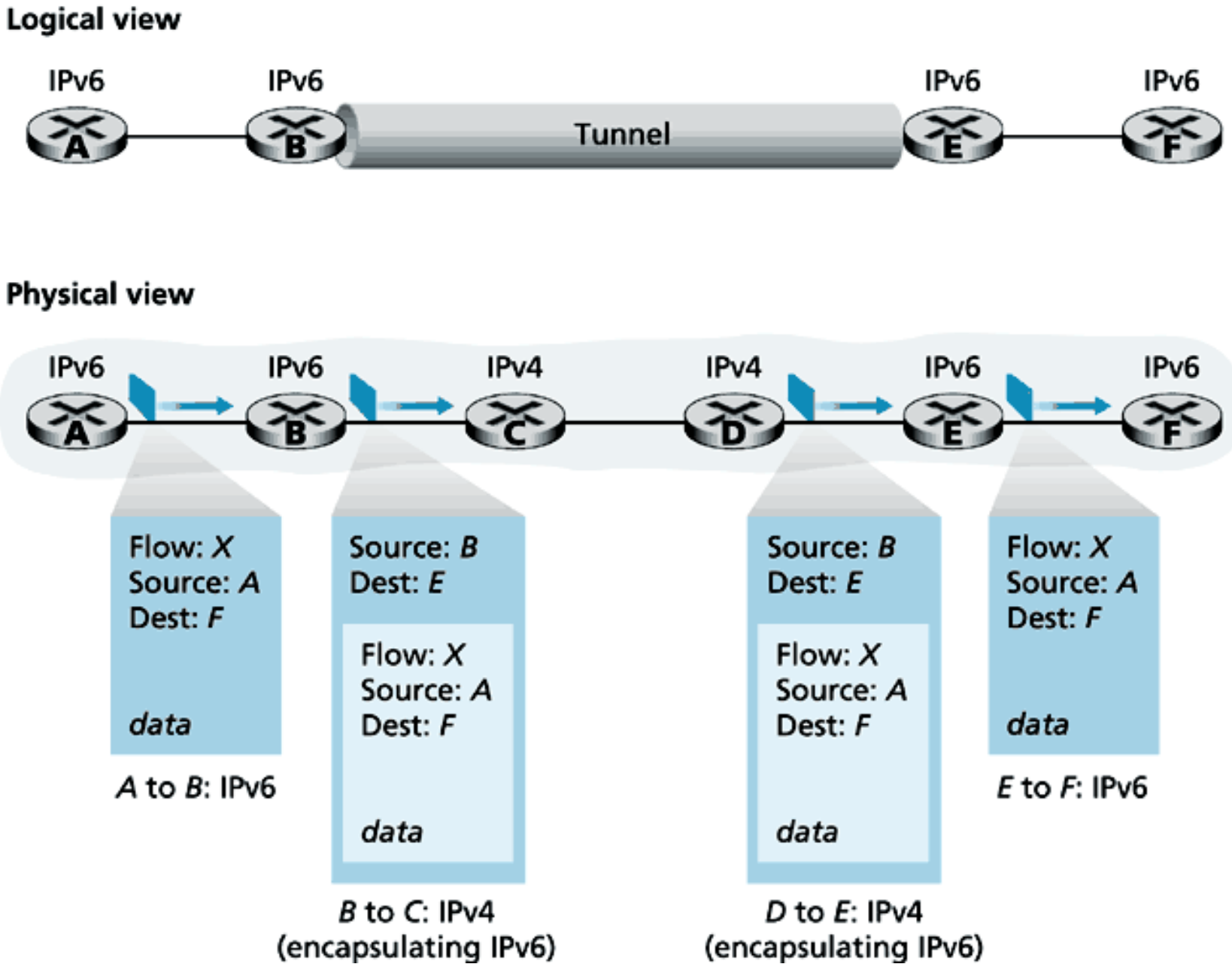
Ejemplo

En la siguiente figura el host de la red privada con IP 10.0.0.1 accede a un servidor Web público con IP 128.119.40.186.



18.6. La transición de IPv4 a IPv6

- RFC 2893.
- A principio de los 90 el IETF predijo que el espacio de direcciones del IPv4 se gastaría aproximadamente en 2008 [14]. Luego apareció el CIDR y se comenzó a hacer un uso intensivo del NAT y del DHCP, con lo que el consumo se ha reducido considerablemente.
- Actualmente existen routers IPv4 e IPv6 funcionando.
- Los routers IPv6 son capaces de encaminar IPv4, pero no al contrario.
- Cuando un router no entiende IPv6 se utiliza una técnica llamada *tunneling*, que consiste en encapsular datagramas IPv6 en datagramas IPv4 cuando los puntos extremos utilizan IPv6 y los routers intermedios sólo entienden IPv4. Ejemplo:


Logical view
IPv6
IPv6
IPv6
IPv6
A
B
Tunnel
E
F
Physical view
IPv6
IPv6
IPv4
IPv4
IPv6
IPv6
A
B
C
D
E
F
Flow: X
Source: A
Dest: F
data
A to B: IPv6
Source: B
Dest: E
Flow: X
Source: A
Dest: F
data
B to C: IPv4
(encapsulating IPv6)
Source: B
Dest: E
Flow: X
Source: A
Dest: F
data
D to E: IPv4
(encapsulating IPv6)
Flow: X
Source: A
Dest: F
data
E to F: IPv6

Capítulo 19

Forwarding (Encaminamiento)

19.1. El proceso de encaminamiento

- Realizado por los routers y en ocasiones los hosts (cuando tienen más de un interface).
- Se realiza para cada paquete a nivel de IP.
- Algoritmo (encaminamiento de un paquete):
  1. Extraer la dir IP destino del paquete.
  2. Buscar la dir IP en la tabla de encaminamiento.
  3. Seleccionar el interface de salida correspondiente.

19.1.1. Búsqueda en las tablas de encaminamiento

- Las tablas de encaminamiento poseen (al menos) los campos:

Red destino	Interface de salida

- La tabla de encaminamiento está indexada por la dirección de red destino.
- En IPv4 existen hasta 2^{32} redes diferentes y en IPv6 hasta 2^{128}. Para evitar tener que disponer físicamente de una entrada para cada posible red, las tablas de encaminamiento normalmente sólo poseen una entrada por cada red que puede alcanzarse directamente desde el nodo.
- Dependiendo del tamaño de dicha red, la entrada tiene más o menos bits.
- Las redes destino pueden ser alcanzadas a través de diferentes interfaces de salida. En este caso, el algoritmo de búsqueda seleccionaría la entrada más larga (la red más pequeña).[1]

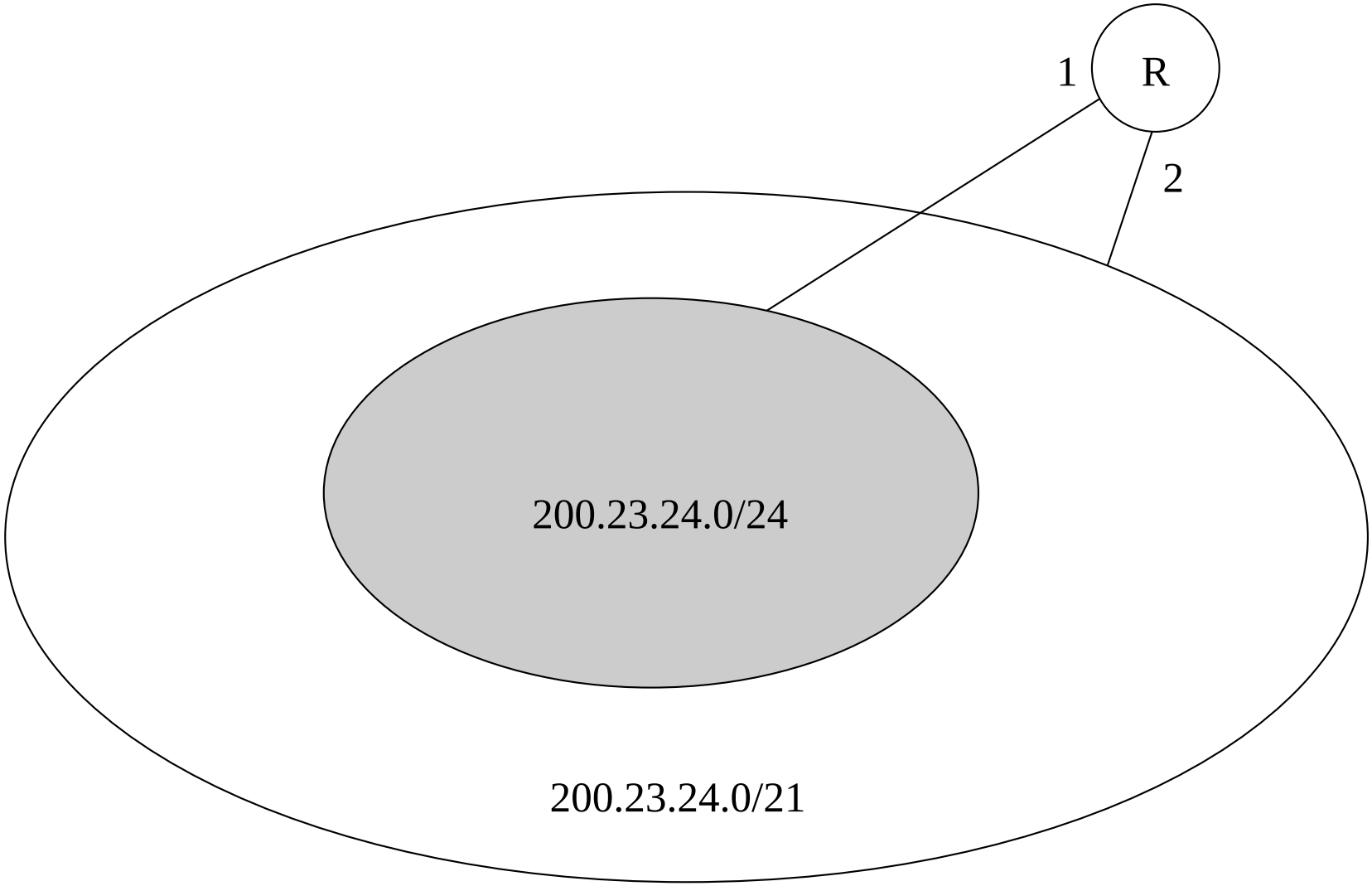
[1] Recuérdese que por definición, unas redes contienen a otras.

Ejemplo

- Un router que posee 4 interfaces y que se encuentra en la red 11001000 00010111 000 (200.23.0.0/19) podría tener una tabla de encaminamiento como la siguiente:

Red de destino	Interface de salida
11001000 00010111 00010 (200.23.16.0/21)	0
11001000 00010111 00011000 (200.23.24.0/24)	1
11001000 00010111 00011 (200.23.24.0/21)	2
en otro caso	3

- Un paquete dirigido a 11001000 00010111 00010xxx xxxxxxxx sería encaminado al interface 0.
- Un paquete dirigido a 11001000 00010111 00011000 xxxxxxxx sería encaminado al interface 1.
- Un paquete dirigido a 11001000 00010111 00011XXX xxxxxxxx sería encaminado al interface 2, si XXX!=000.


1
R
2
200.23.24.0/24
200.23.24.0/21

Ejemplo

En Unix, la tabla de encaminamiento del host se puede consultar con el comando:

```
$ /sbin/route
Kernel IP routing table
Destination   Gateway          Genmask         Iface
193.147.118.0 *                255.255.255.0 eth0
loopback      gogh.ace.ual.es 255.0.0.0     lo
default       193.147.118.1   0.0.0.0         eth0
```

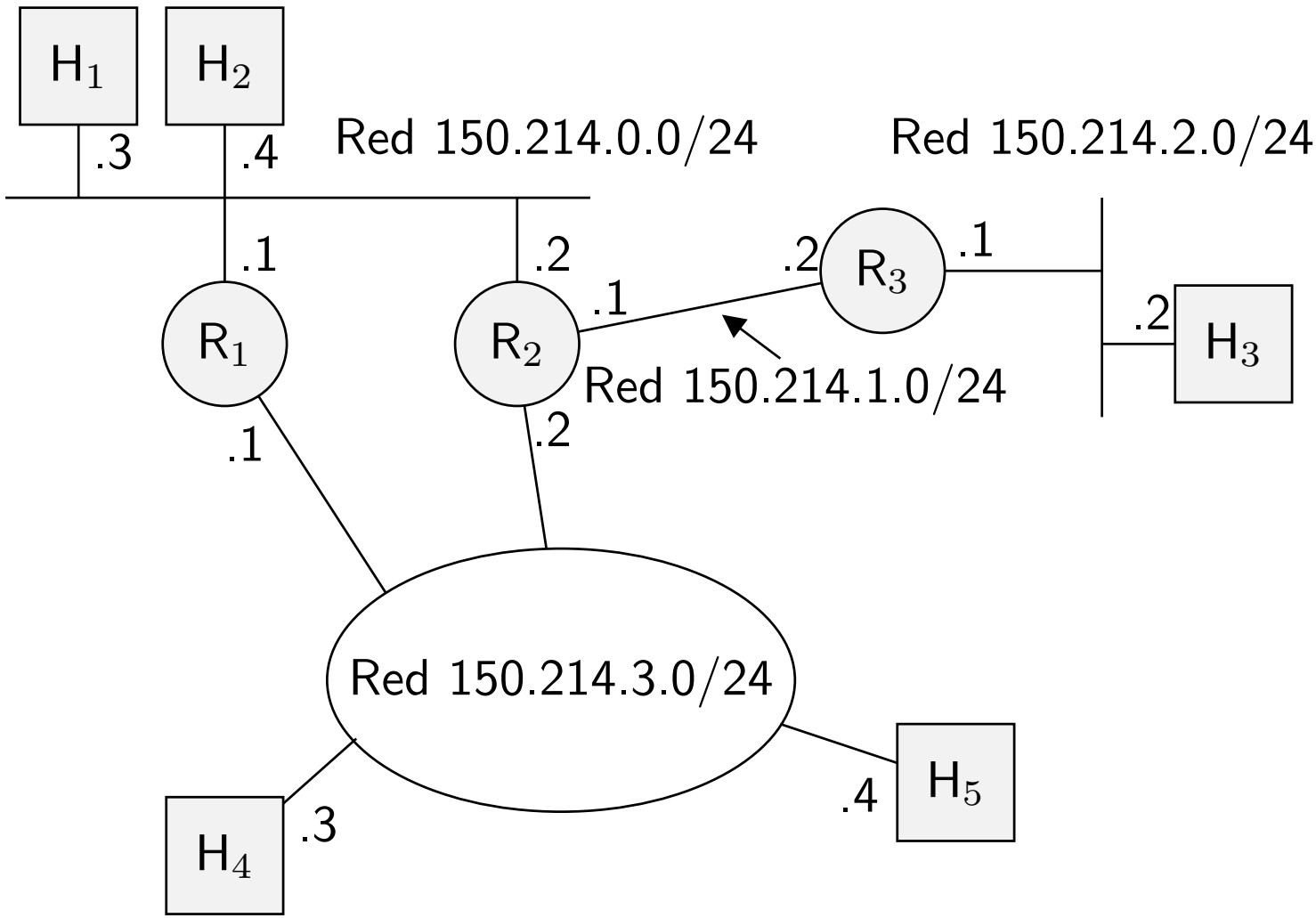
Según dicha tabla (del host `gogh.ace.ual.es`):

- Cualquier paquete que vaya dirigido a una estación de la red local (que en este ejemplo se trata de una red de clase C) debe ser entregado al interface de red `eth0`.
- Cualquier paquete que vaya dirigido al `loopback` (una red virtual de clase A formada sólo por el host) debe ser entregado al interface de red `lo`.

- Finalmente, cualquier paquete que no vaya dirigido ni a la red local ni al `loopback`, será entregado al interface de red `eth0`.

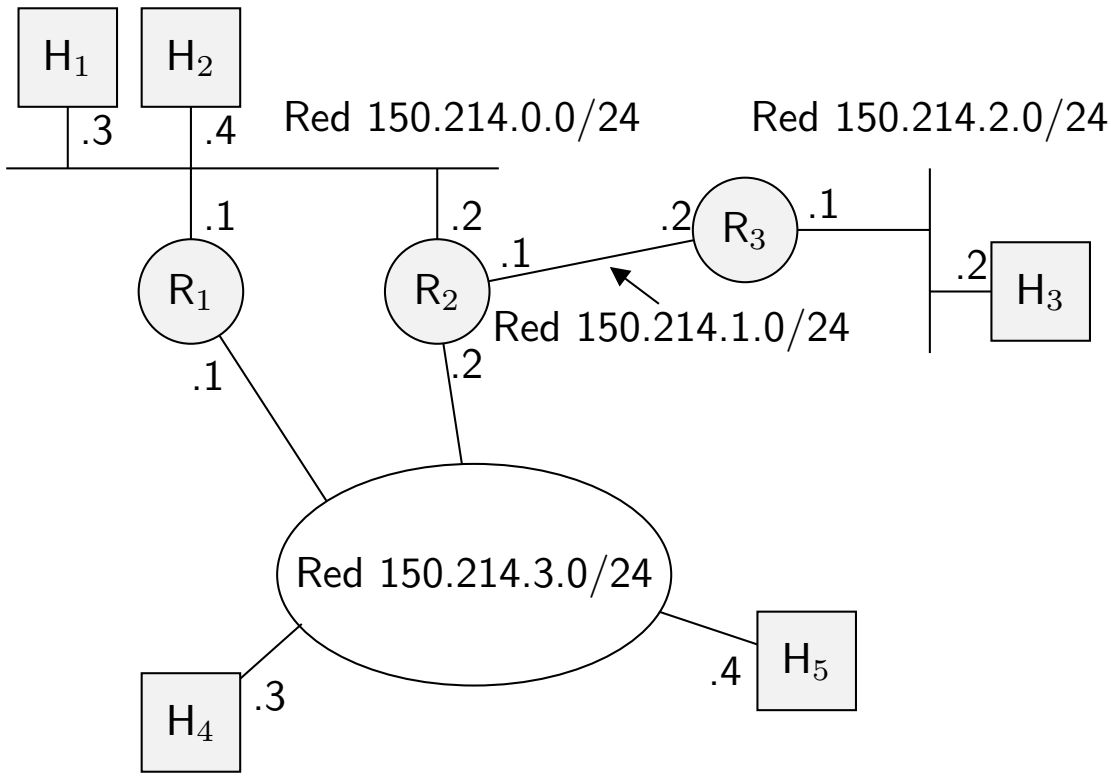
Ejemplo

El router R_2 de la red



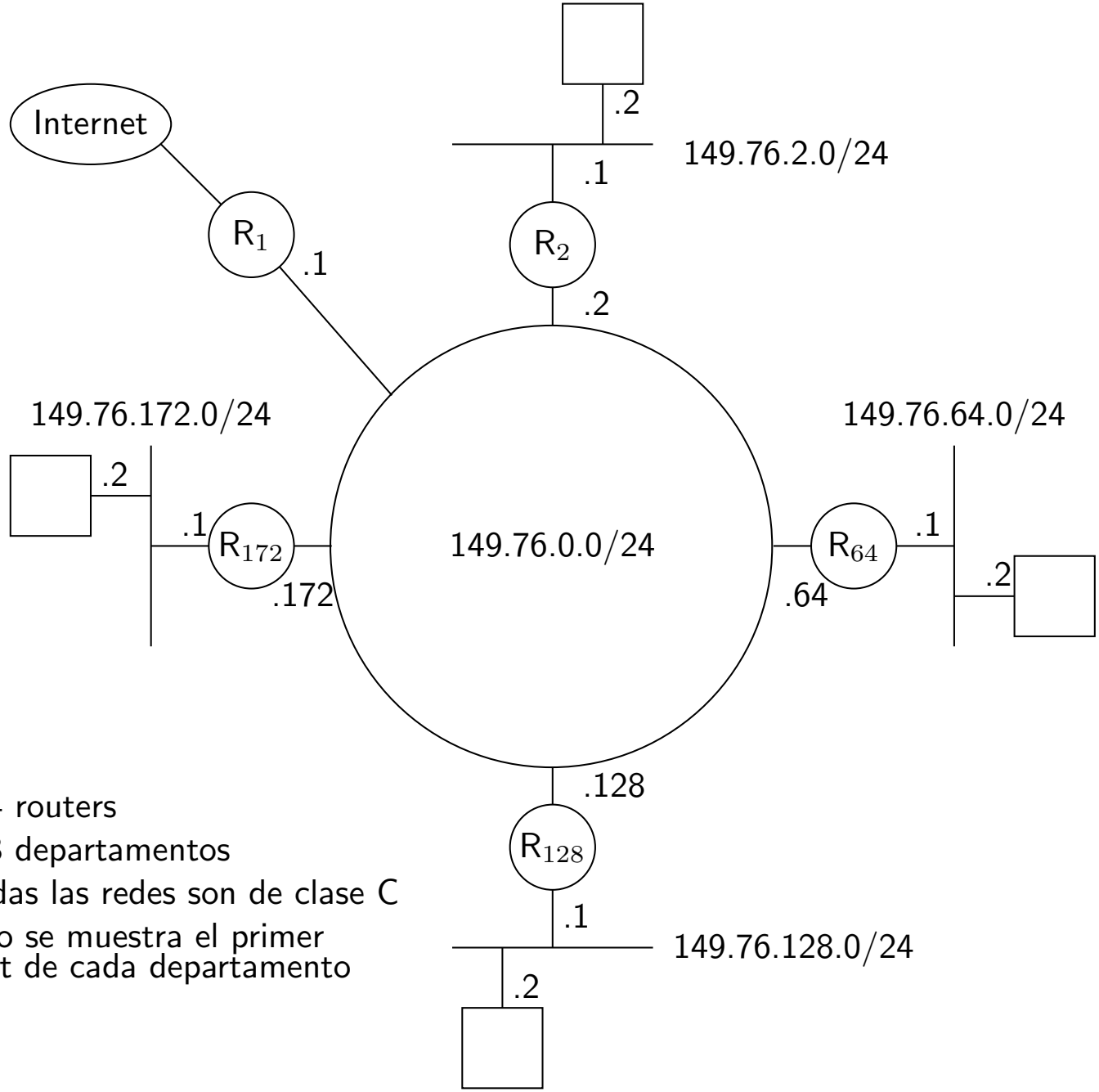
podría tener una tabla de encaminamiento igual a:

Destination	Gateway	Genmask	Iface
150.214.0.0	*	255.255.255.0	eth0
150.214.1.0	*	255.255.255.0	ppp0
150.214.3.0	*	255.255.255.0	fddi0
loopback	R2	255.0.0.0	lo
default	150.214.1.2	0.0.0.0	ppp0



Ejemplo

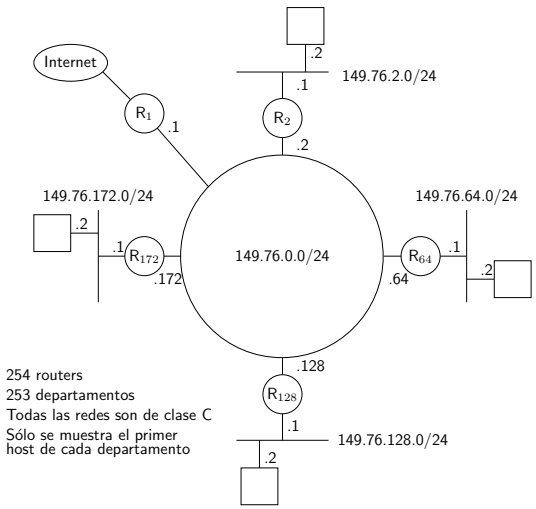
Supongamos una red clase B con dirección 149.76.0.0 dedicada a un campus universitario. Debido a su excesivo tamaño (2^{16} interfaces/red), esta red se divide en redes clase C (2^8 interfaces/red). Físicamente se distribuye una dorsal (backbone) de FDDI y a ella se conectan las pasarelas (gateways) que unen la FDDI con las redes Ethernet de los diferentes departamentos. ¿Cómo administrarías las direcciones de red y qué direcciones asignarías a las pasarelas? ¿Cuántos departamentos de hasta 254 estaciones pueden formarse? Especifique la tabla de encaminamiento para la pasarela 149.76.0.4 suponiendo que todas las redes posibles del campus están dadas en alta en ella.



254 routers
253 departamentos
Todas las redes son de clase C
Sólo se muestra el primer host de cada departamento

El router 149.74.0.4 posee la siguiente tabla de routing:

Destino	Router	Máscara	Interface
149.76.2.0	149.76.0.2	255.255.255.0	fddi0
149.76.3.0	149.76.0.3	255.255.255.0	fddi0
149.76.4.0	*	255.255.255.0	eth0
149.76.5.0	149.76.0.5	255.255.255.0	fddi0
⋮	⋮	⋮	⋮
149.76.254.0	149.76.0.254	255.255.255.0	fddi0
127.0.0.0	*	255.0.0.0	lo
default	149.76.0.1	0.0.0.0	fddi0



19.2. Agregación de direcciones

- Cuando dos (o más subredes) poseen el mismo prefijo, pueden agregarse formando una red mayor (véase el ejemplo de la página 363).
- Dicha agregación hace posible reducir el tamaño de las tablas de encaminamiento (eliminando en dicho ejemplo la entrada de la red más larga que está incluida en la otra entrada).
- La agregación no significa que las redes se fusionen. La agregación sólo afecta a las tablas de encaminamiento.

Ejemplo de agregación

- Sea la siguiente configuración, donde existen dos ISP's que conectan a una serie de redes a Internet. El primer ISP Fly-By-Night da servicio a 8 organizaciones y el segundo ISP ISPs-R-Us hace lo mismo a un número indeterminado de ellas.
- Las redes conectadas a través de Fly-By-Night poseen un prefijo común igual a 200.23.16.0/20. Las redes conectadas a través de ISPs-R-Us tienen todas el prefijo 199.31.0.0/16.
- Gracias a la agregación, el router que conecta Internet con Fly-By-Night sólo posee una entrada en su tabla de encaminamiento con la red destino 200.23.16.0/20, aunque en realidad existen 8 redes dentro de esta. Lo mismo ocurre con el router que conecta a ISPs-R-Us con Internet.

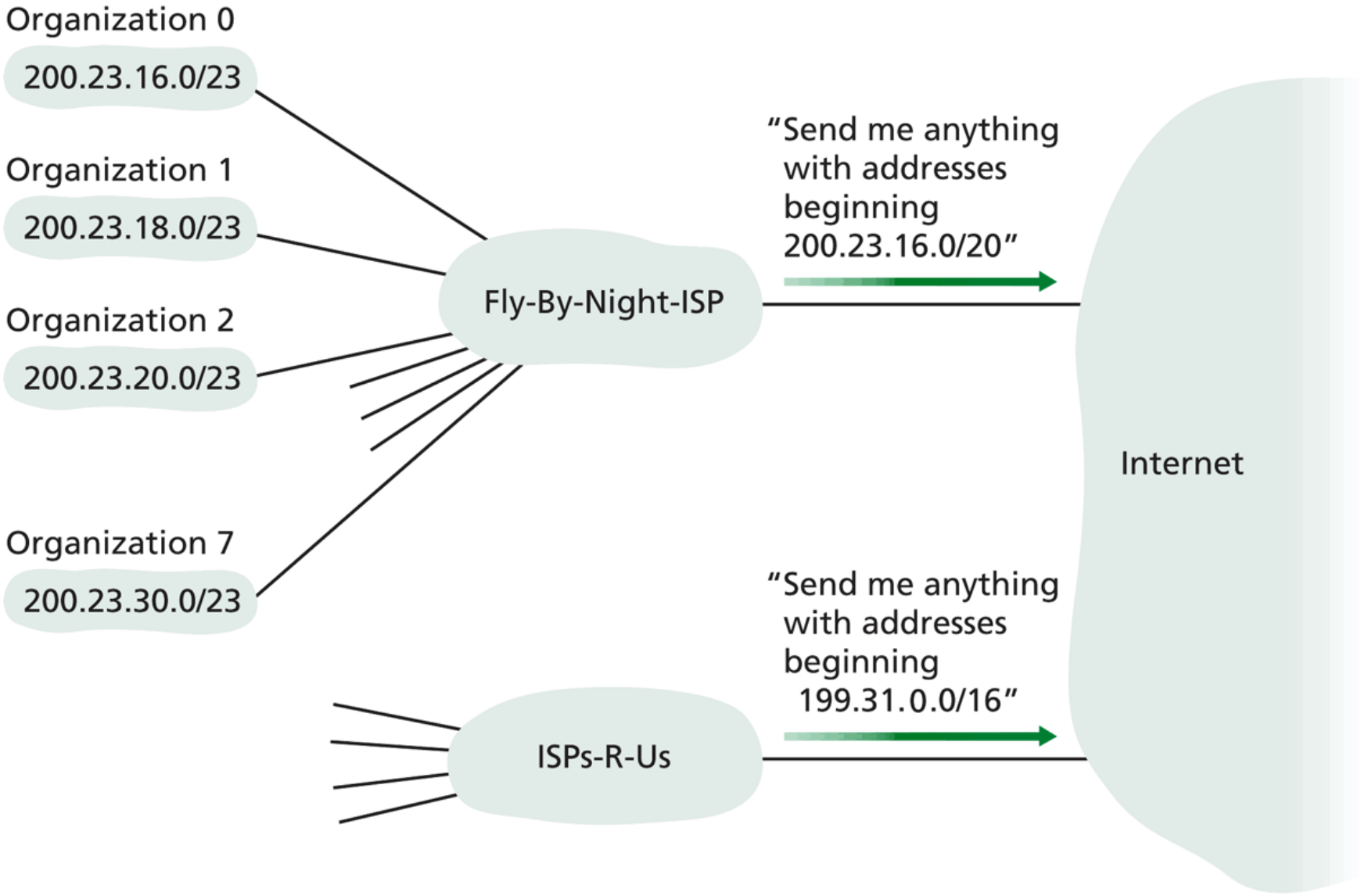


Figure 4.18 ♦ Hierarchical addressing and route aggregation

Ejemplo de des-agregación

- Imaginemos que Fly-By-Night no es un ISP muy serio y que la Organización 1 se pasa a ISPs-R-Us. Sin embargo, la Organización 1 no desea modificar sus dirs IP.
- Esta nueva configuración debe afectar a los routers que conectan ambos ISP's con Internet para hacer que los paquetes dirigidos a la Organización 1 sean encaminados a través de ISPs-R-Us.
- Supongamos (por simplicidad) que Fly-By-Night e ISPs-R-Us se conectan a Internet a través del mismo router. En la tabla de encaminamiento de este router figurará una entrada para la red 200.23.16.0/20 y otra para la red 200.23.18.0/23.
- Como 200.23.16.0/20 es prefijo de 200.23.18.0/23, cada vez que a este router lleque un paquete dirigido a la Organización 1, usará el prefijo más largo (200.23.18.0/23) y el encaminamiento será correcto. Más específicamente, su tabla de encaminamiento sería:

Red de destino	Interface de salida
⋮	⋮
199.31.0.0/16	El que lleva a ISPs-R-Us
200.23.18.0/23	El que lleva a ISPs-R-Us
200.23.16.0/20	El que lleva a Fly-By-Night
⋮	⋮

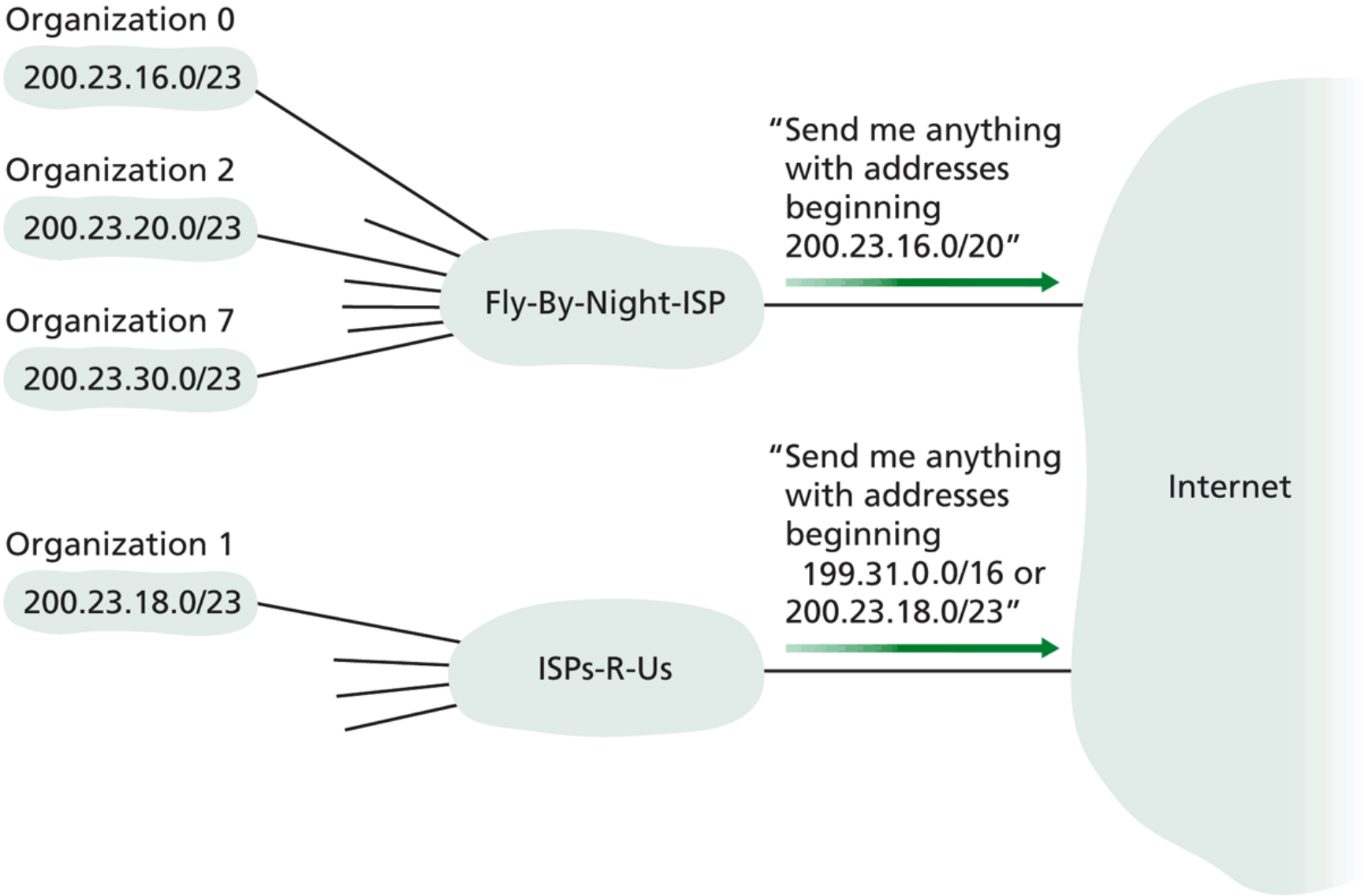


Figure 4.19 ♦ ISPs-R-Us has a more specific route to Organization 1

Capítulo 20

Routing (Rutado)

20.1. ¿Qué es el routing?

- Si forwarding consiste en transmitir un paquete hasta el siguiente nodo, routing es el proceso de determinar el mejor camino para realizar el encaminamiento. En otras palabras, routing es el proceso que se realiza para determinar las tablas de encaminamiento.
- El problema del routing se trata generalmente modelando las redes mediante grafos. En ellos, los nodos representan a los routers de la red y los arcos a los enlaces que los interconectan.

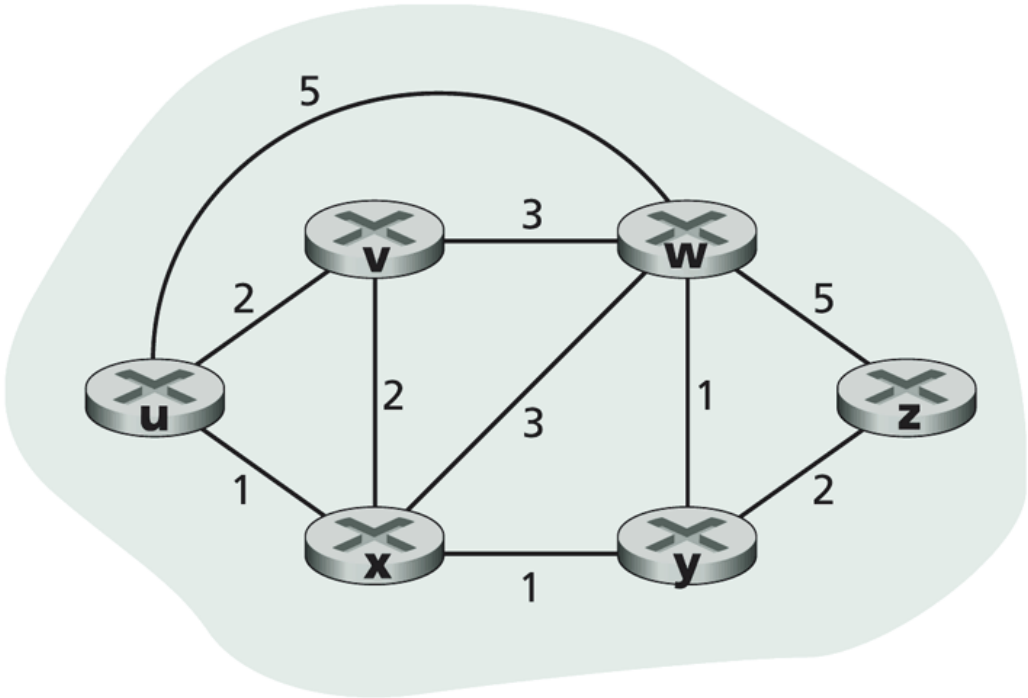


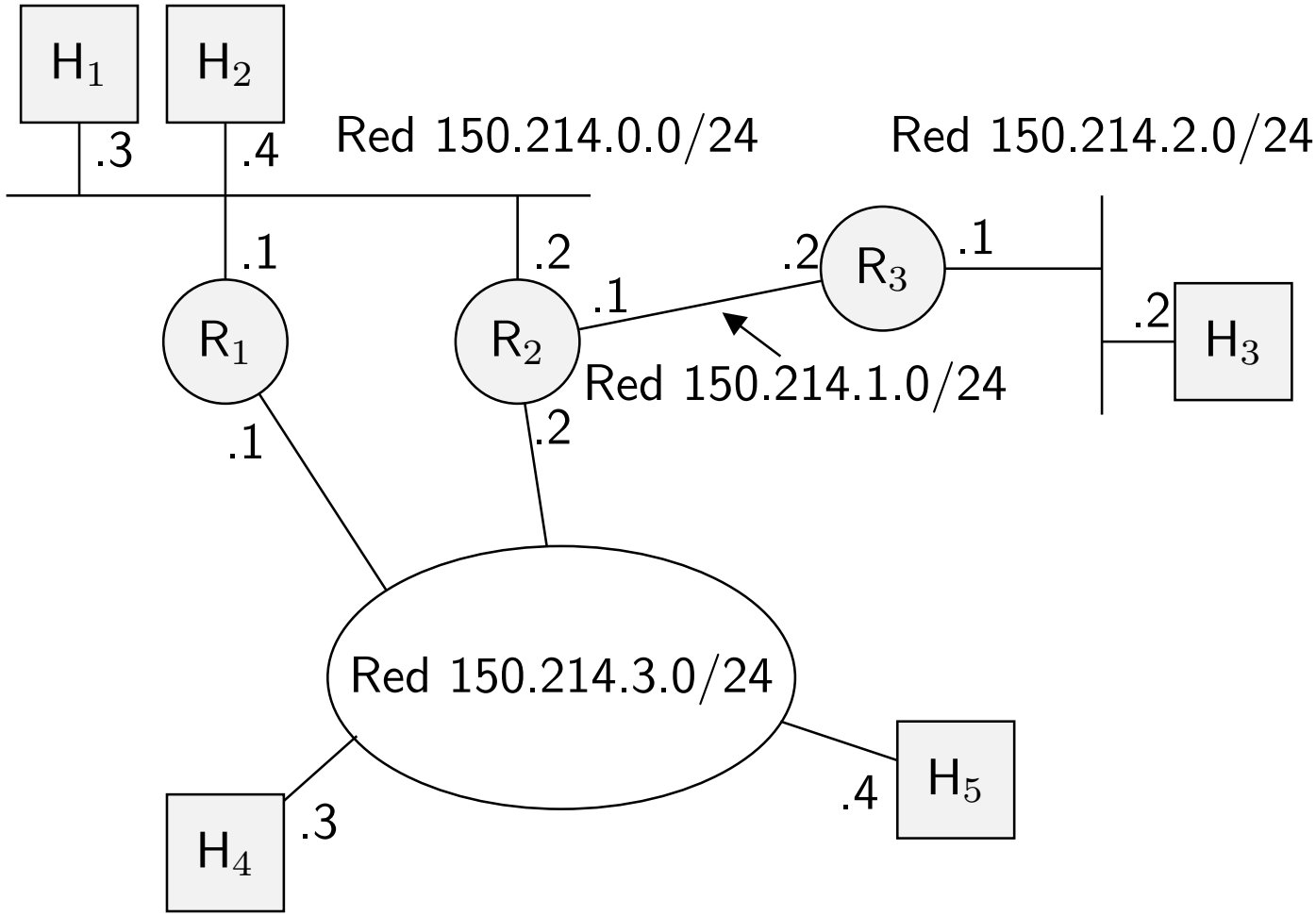
Figure 4.25 ♦ Abstract graph model of a computer network

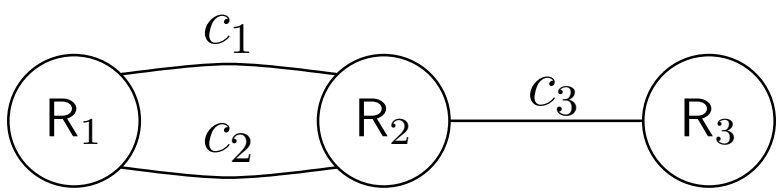
- La solución más frecuente que buscan los algoritmos de routing consiste en encontrar el camino más corto que une a cada par de routers (el router origen del paquete y el router destino). Como es lógico, el router origen es en realidad el *first-hop router* (el que conecta con Internet a la red que contiene el host que en realidad genera el paquete) y el router destino es el *last-hop router* (el que conecta con Internet

a la red que contiene el host que es el destino del paquete).

- Evidéntemente, ninguno de estos routers son realmente ni el origen ni el destino del paquete. En realidad son las redes que se conectan a estos routers. Sin embargo, el problema del routing queda resuelto si obviamos este hecho porque independientemente de la red destino a la que finalmente va dirigido el paquete, este debe de llegar forzosamente al *last-hop router*, y el mismo razonamiento puede realizarse para el *first-hop router*.

20.2. La red como un grafo



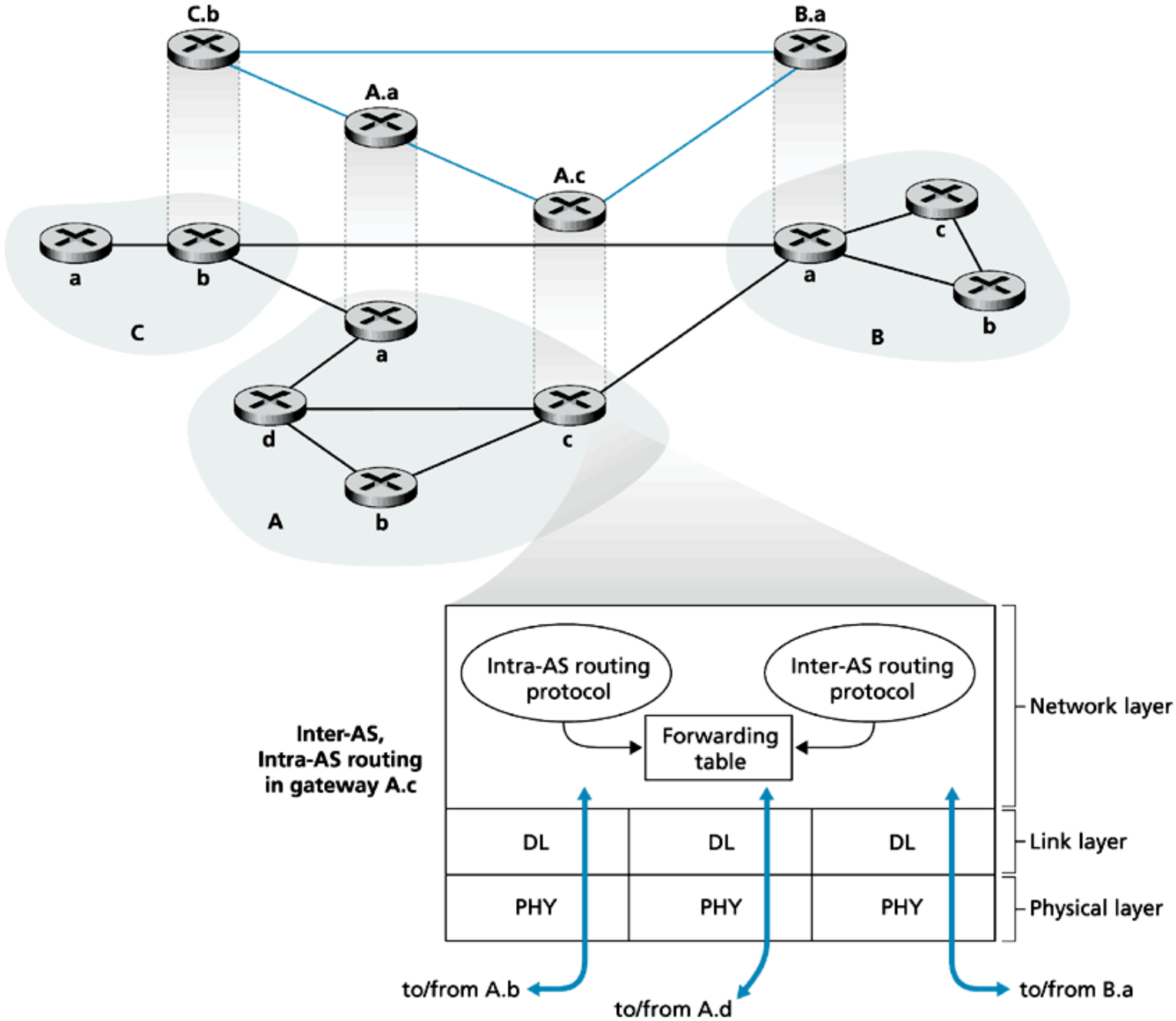


20.3. Sobre los costes de los caminos

- Se utilizan para determinar las rutas óptimas.
- Dependen de:
  1. Latencia de los enlaces (distancia física entre nodos).
  2. Tasa de transmisión (capacidad) de los enlaces.
  3. Carga de los enlaces.
  4. Tasa de errores de los enlaces.
  5. Cuestiones políticas (espionaje) y/o comerciales (coste económico).

20.4. Routing jerárquico y sistemas autómomos

- Internet está formada por una estratificación de sistemas autónomos o AS's (Autonomous Systems) de diferente nivel que son administrados de forma independiente.
- Dentro de cada sistema autónomo se realiza el routing como se considera necesario, sin tener en cuenta lo que ocurre en el resto de AS's. Lo único que debe verificarse es que el protocolo de routing que ejecutan los routers del AS debe de ser el mismo (para que se entiendan entre sí).
- En la siguiente figura se muestra un ejemplo de varios AS interconectados entre sí. En cada AS existen routers que ejecutan un protocolo de routing intra-AS y al menos 1 que ejecuta un protocolo inter-AS. Gracias al protocolo intra-AS los routers de cada AS saben cómo encaminar dentro de su AS y gracias al protocolo inter-AS (al menos) uno de los routers de cada AS sabe cómo encaminar entre AS's.


C.b
B.a
A.a
A.c
a
b
C
a
d
c
A
b
a
c
b
B
Inter-AS,
Intra-AS routing
in gateway A.c
Intra-AS routing protocol
Inter-AS routing protocol
Forwarding table
Network layer
DL
DL
DL
Link layer
PHY
PHY
PHY
Physical layer
to/from A.b
to/from A.d
to/from B.a

- Por tanto, el routing jerárquico permite esconder la complejidad a de los AS's de nivel inferior a los AS's de nivel superior. De esta manera:

  1. Las tablas de encaminamiento son mucho más pequeñas. De hecho, un router de un AS sólo necesita conocer los routers de su AS.
  2. El número de mensajes intercambiados por los routers es mucho menor y la mayoría se producen dentro de los AS's. Así los algoritmos de routing dinámicos convergen más rápidamente.
  3. Permite la autonomía administrativa (algoritmo de routing, políticas, etc.).

20.5. El RIP (Routing Information Protocol)

- RFC's 1058 y 2453.
- Routing intra-AS (interior gateway protocol [15]).
- Creado por Xerox e incluído en la versión BSD (Berkeley Software Distribution) del UNIX en 1982.
- Los routers utilizan el algoritmo de routing *Distance-Vector*.
- El coste de cada enlace es siempre 1. Por tanto, el algoritmo minimiza el número de saltos (hops).
- El coste máximo permitido para un camino (path) es 15 (AS's pequeños).
- Los routers se intercambian (entre vecinos inmediatos) sus "tablas de routing" (vectores con las distancias a todas las redes del AS y routers por los que encaminar hacia ellas) cada 30 segundos (no necesariamente de forma síncrona) y cuando reciben nuevos datos acerca del AS, re-calculan los caminos mínimos.

- Si un router tras el cálculo detecta alguna variación en su tabla de routing, esta es comunicada a todos sus vecinos directamente conectados.
- Se utilizan paquetes UDP y el puerto 520 [14].
- Transcurridos 180 segundos sin recibir información desde el router X, todo router conectado directamente a X lo considera inalcanzable (unreacheable). En este momento re-calcula los caminos mínimos (teniendo en cuenta este evento) y transmite la tabla de routing con los nuevos caminos a todos sus vecinos.
- El número máximo de redes destino en cada mensaje es a lo sumo 25 [15].

- El RIP es utilizado por el demonio "routed" que corre, en la capa de aplicación y como un proceso más, en el seno del sistema operativo UNIX y compatibles.

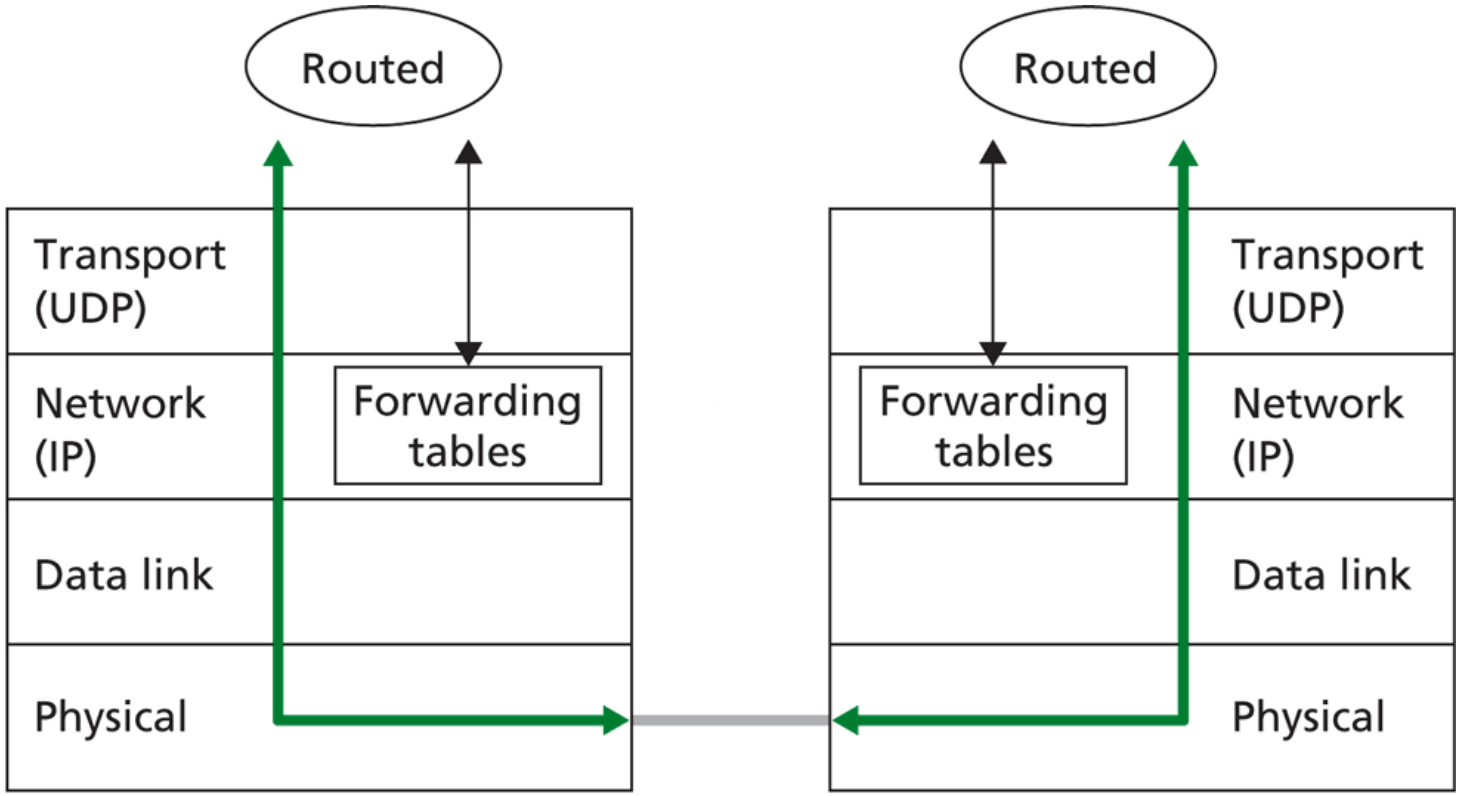


Figure 4.36 ♦ Implementation of RIP as the *routed* daemon

Ejemplo

Sea el SA de la figura:

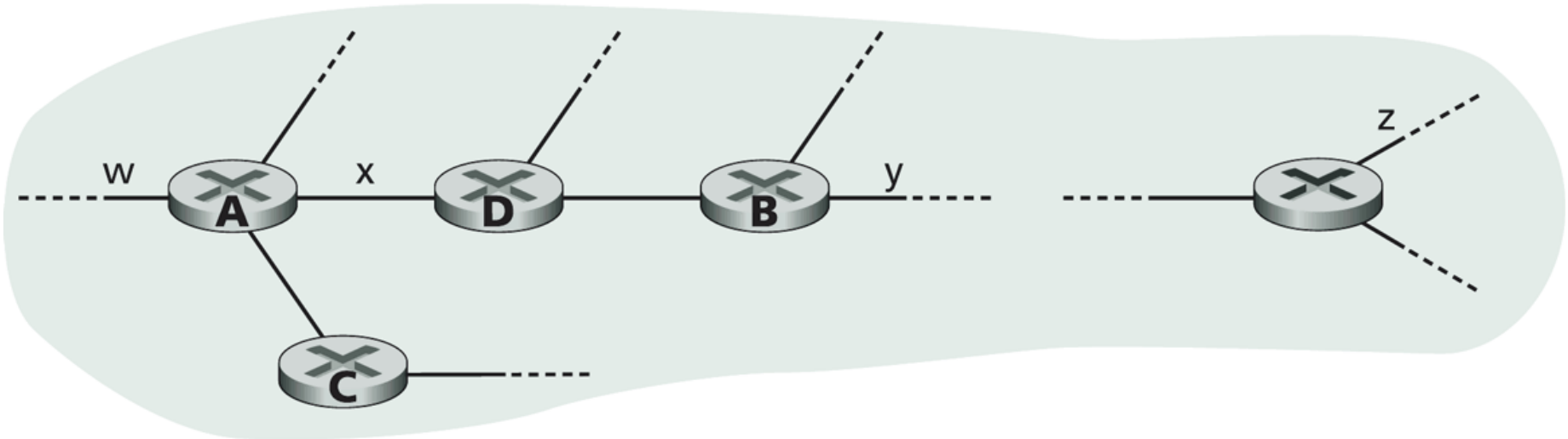


Figure 4.32 ♦ A portion of an autonomous system

En un momento determinado la tabla de routing del router D podría ser:

Destination subnet	Next router	Number of hops to destination
w	A	2
y	B	2
z	B	7
x	-	1
⋮	⋮	⋮

Si más tarde recibe la siguiente tabla de routing de A:

Destination subnet	Next router	Number of hops to destination
z	C	4
w	-	1
x	-	1
⋮	⋮	⋮

Entonces la tabla de routing del router D pasaría a ser:

Destination subnet	Next router	Number of hops to destination
w	A	2
y	B	2
z	A	5
x	-	1
⋮	⋮	⋮

20.6. El protocolo OSPF (Open Shortest Path First)

- RFC 2328.
- Routing intra-AS.
- Se diseñó como el sucesor del RIP y puede manejar AS más grandes.
- Los intercambios de los mensajes con información acerca del routing son autentificados.
- Los routers utilizan el algoritmo de routing *Link-State* para calcular los caminos de coste mínimo.
- Los costes de los enlaces no tienen que ser siempre 1 como en el RIP. Así, el administrador de la red podría tener en cuenta la capacidad de los enlaces, por ejemplo.

- El cálculo de los caminos mínimos se produce cada vez que un router detecta un cambio en el coste de uno de sus enlaces o en su defecto, cada 30 minutos. En cualquier caso, los routers envian a todos los demás routers del AS (broadcasting) los costes de los enlaces con dichos routers.
- Se utiliza directamente el IP, puerto 89 [14].
- Soporta multicasting (RFC 1584).

- Soporta routing jerárquico (permite definir AS's dentro de los AS's):

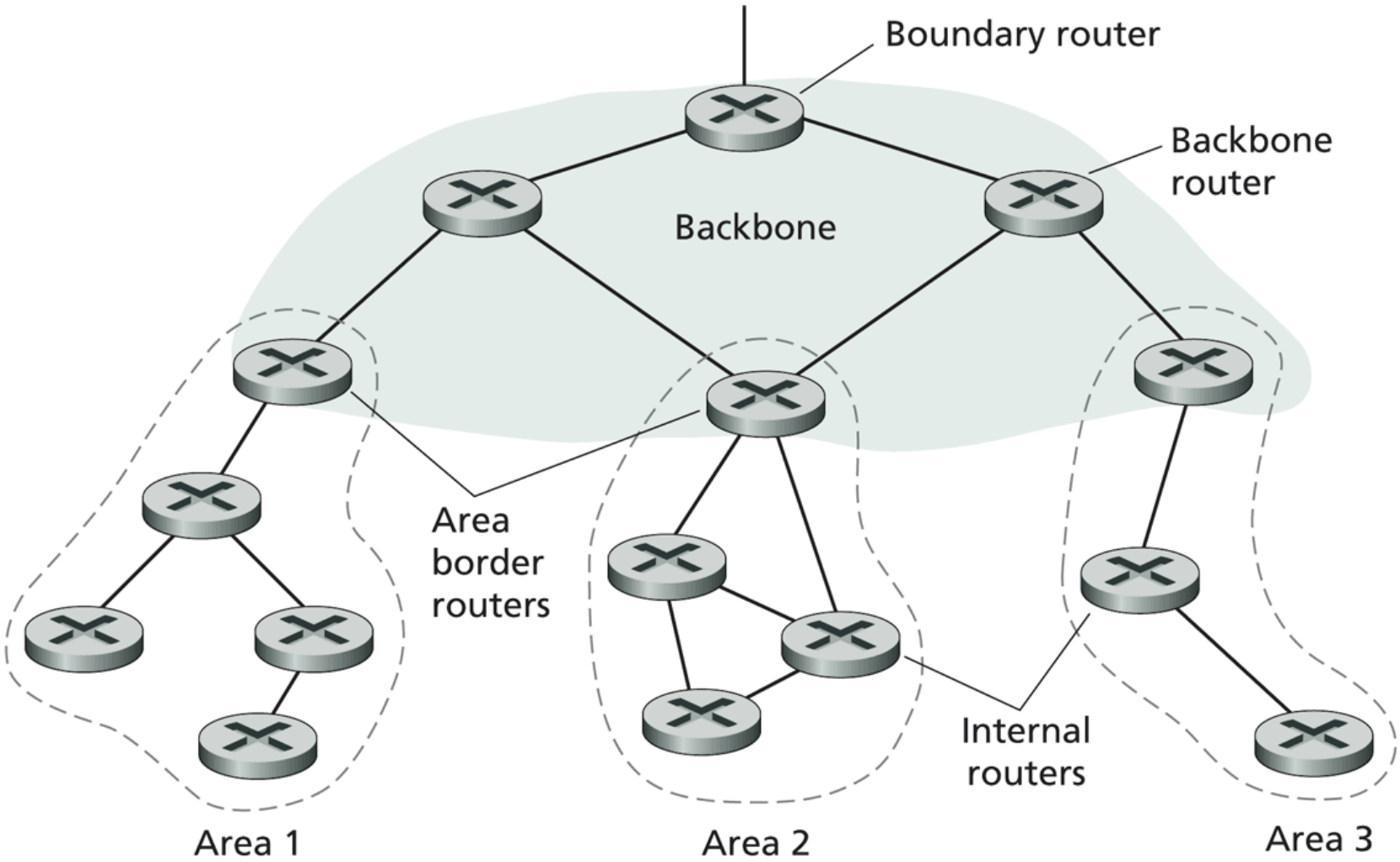
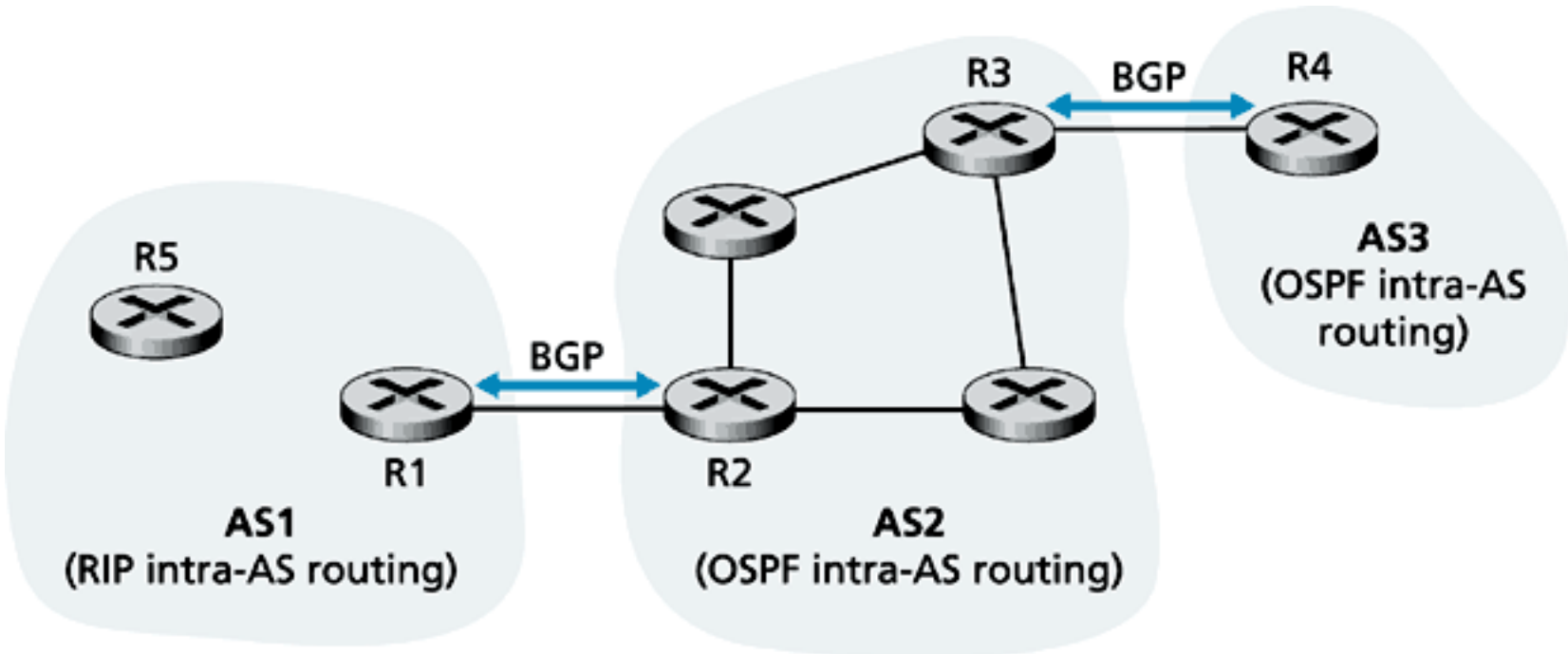


Figure 4.37 ♦ Hierarchically structured OSPF AS with four areas

20.7. El BGP (Border Gateway Protocol)

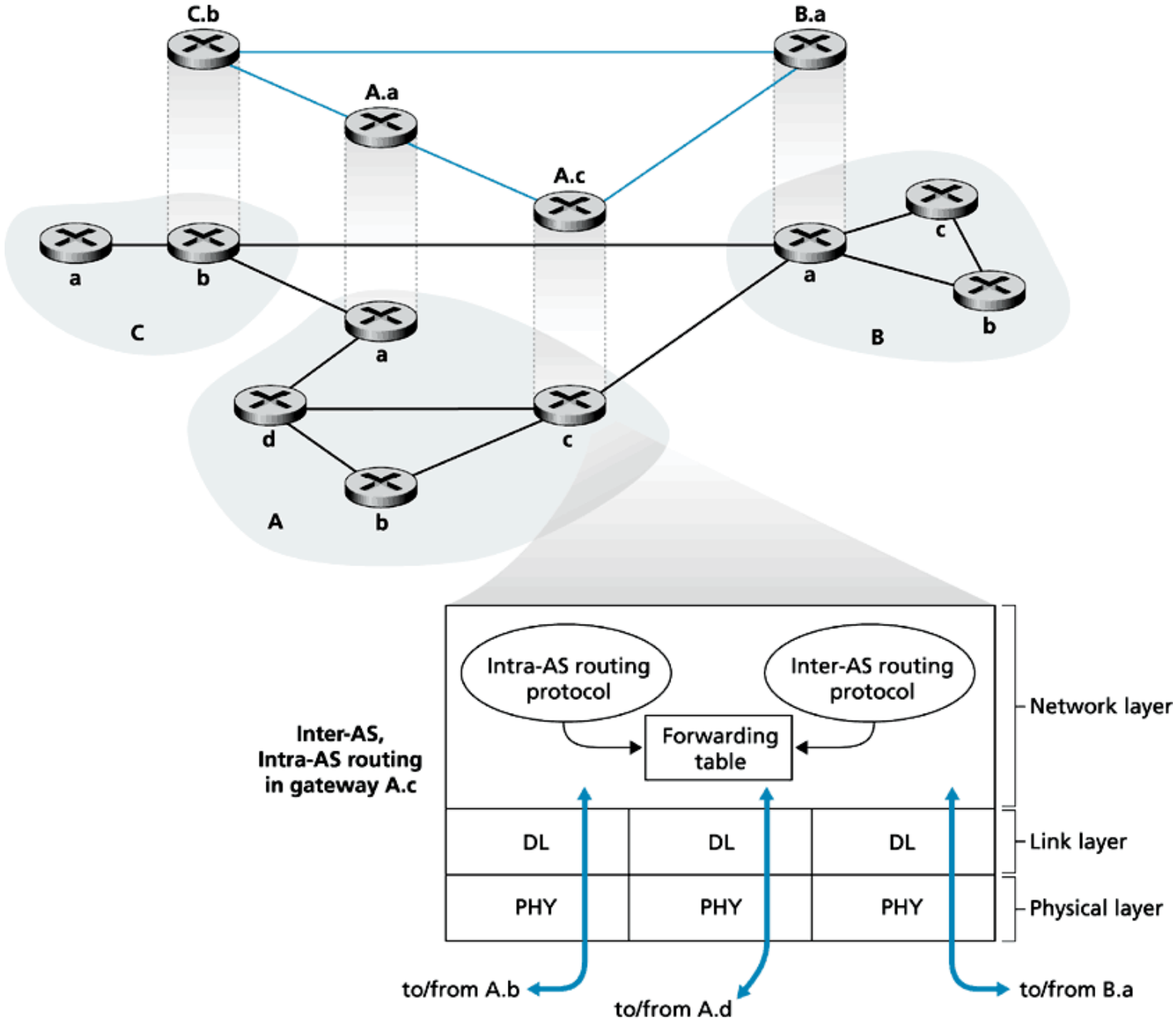
- RFC's 1771 (versión 4), 1172, 1173 y 1930.
- Routing inter-AS.[1]



- Utiliza el algoritmo Distance-Vector.

[1] A los routers que conectan sistemas autónomos entre sí se les llama border gateways.

- BGP usa intensivamente el routing jerárquico.
- A la hora de calcular las rutas se tienen en cuenta factores económicos, políticos, de seguridad, etc.
- Se emplea el TCP, puerto 179.
- Los routers se intercambian *CDIRized prefixes* (direcciones de redes que contienen otras redes de forma jerárquica) [15]. Por ejemplo (ver la siguiente figura), el router A.c enviará al router B.a información sobre las redes que son alcanzables desde A.c y B.a enviará a A.c información sobre las redes que son alcanzables desde B.a. Nótese que si la agregación es óptima, el intercambio de información es mínimo:


C.b
B.a
A.a
A.c
a
b
C
a
d
c
A
b
a
c
b
B
Inter-AS,
Intra-AS routing
in gateway A.c
Intra-AS routing protocol
Inter-AS routing protocol
Forwarding table
Network layer
DL
DL
DL
Link layer
PHY
PHY
PHY
Physical layer
to/from A.b
to/from A.d
to/from B.a

20.8. Algoritmos de routing

20.8.1. Algoritmo de routing Link-State

- Consta de dos fases:

  1. Fase de flooding (inundación). En ella, cada router "*broadcasts*" el estado de sus enlaces (destino[2] y coste) al resto de routers del AS.
  2. Fase de cálculo de los caminos mínimos. Se utiliza normalmente el algoritmo de Dijkstra [22].

[2] Nótese que el origen está determinado por quien envía en mensaje.

Flooding de los estados de los enlaces

- Cada router envía el estado de sus enlaces a todos los routers directamente conectados.
- Todo router que recibe un mensaje con información del tipo anterior retransmite el mensaje a través de todos sus enlaces excepto por el que llegó el mensaje.
- Cuando los mensajes son creados se les asigna un TTL que es decrementado cada vez que es retransmitido. De esta forma garantizamos que la fase de inundación finaliza.

Algoritmo de Dijkstra

1. Sean dos listas (*Confirmed* y *Tentative*) de ternas (*Destination*, *Cost*, *NextHop*). Inicializar *Confirmed* con el nodo que ejecuta el algoritmo y *Tentative* con los nodos que pueden alcanzarse desde él.

2. Mientras *Tentative* no esté vacía:

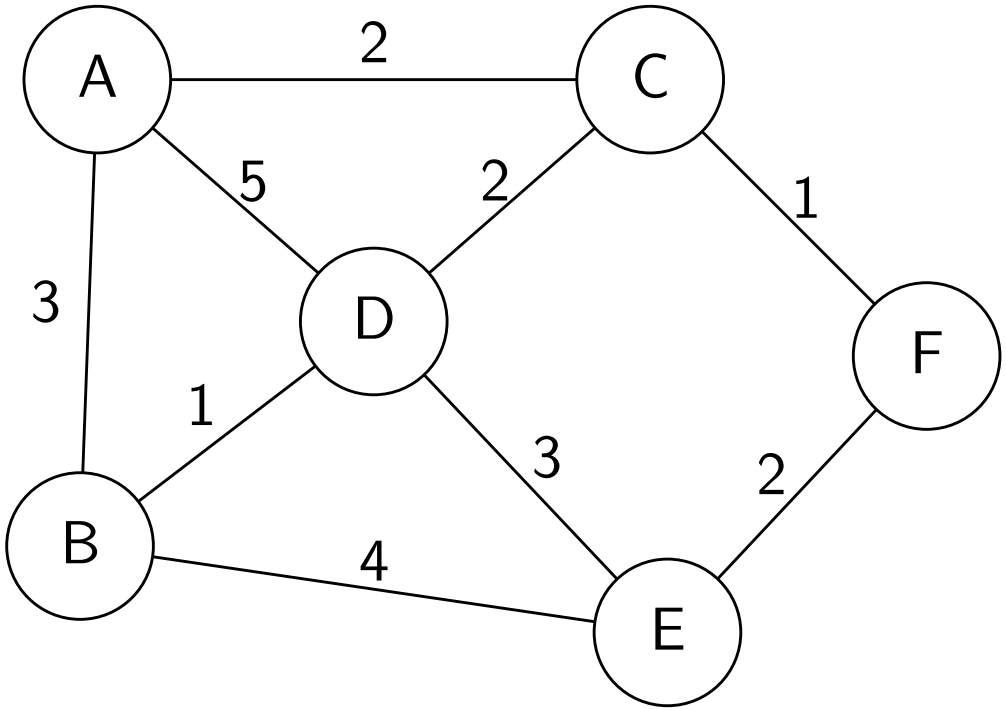
   a) Mover desde *Tentative* a *Confirmed* la terna de menor coste.

   b) Recarcular las distancias a los posibles destinos con esta terna.

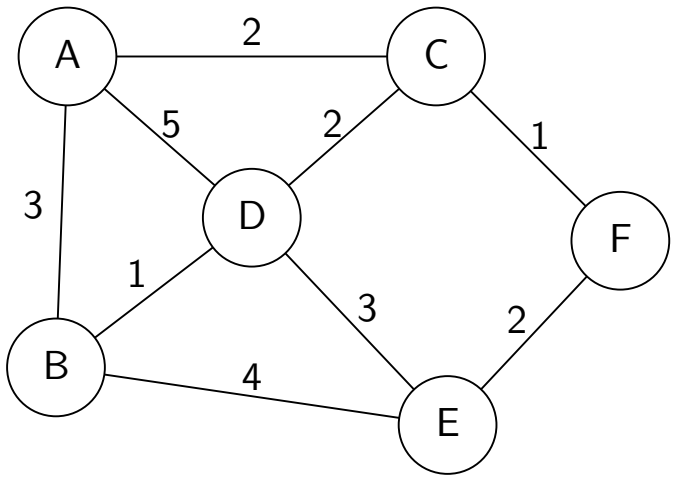
3. En cada entrada de *Confirmed* queda almacenada la distancia mínima a cada nodo de la red y el primer nodo que se tiene que atravesar para conseguirlo.

Ejemplo

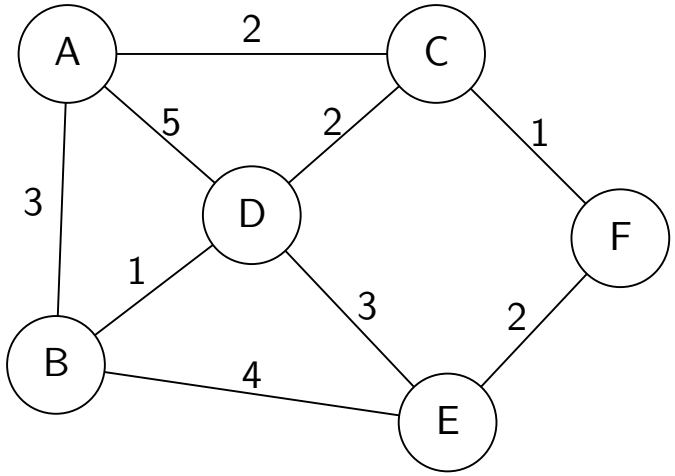
- Dada la red de la figura



obtener la tabla de encaminamiento para el nodo E según el algoritmo Link-State.



Confirmed	*Tentative*	Comentarios
(E,0,-)	(B,4,B) (D,3,D) (F,2,F)	Inicio
(E,0,-) (F,2,F)	(B,4,B) (D,3,D)	Movemos (F,2,F)
(E,0,-) (F,2,F)	(B,4,B) (D,3,D) (C,2+1,F)	Actualizamos desde (F,2,F)
(E,0,-) (F,2,F) (D,3,D)	(B,4,B) (C,3,F)	Movemos (D,3,D)
(E,0,-) (F,2,F) (D,3,D)	(B,4,B) (C,3,F) (A,3+5,D)	Actualizamos desde (D,3,D)
(E,0,-) (F,2,F) (D,3,D) (C,3,F)	(B,4,B) (A,8,D)	Movemos (C,3,F)
(E,0,-) (F,2,F) (D,3,D) (C,3,F)	(B,4,B) (A,3+2,F)	Actualizamos desde (C,3,F)



Confirmed	*Tentative*	Comentarios
(E,0,-) (F,2,F) (D,3,D) (C,3,F) (B,4,B)	(A,5,F)	Movemos (B,4,B)
(E,0,-) (F,2,F) (D,3,D) (C,3,F) (B,4,B)	(A,5,F)	Actualizamos desde (B,4,B)
(E,0,-) (F,2,F) (D,3,D) (C,3,F) (B,4,B) (A,5,F)		Movemos (A,5,F) y terminamos

20.8.2. Algoritmo de routing Distance-Vector

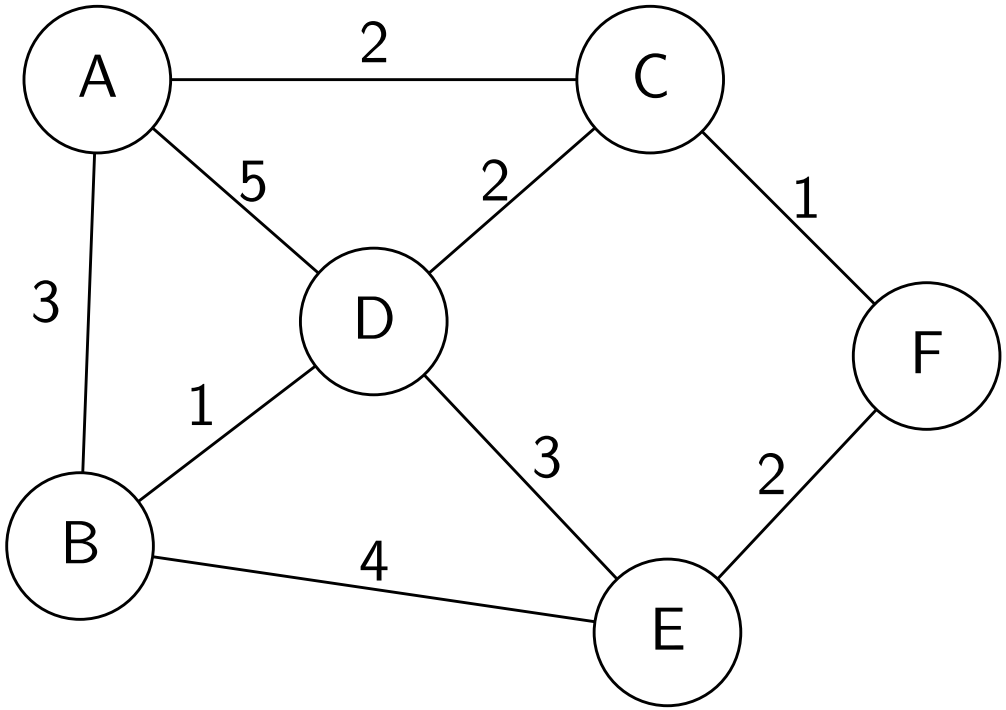
- También llamado algoritmo de Bellman-Ford [22].
- Emplea el mismo modelo de red mediante un grafo, donde los nodos representan routers y los arcos enlaces de transmisión.
- Es un algoritmo progresivo (calcula las rutas progresivamente, mejorándolas en cada iteración) y distribuido (los cálculos se realizan parcialmente en cada router).

Algoritmo de Bellman-Ford

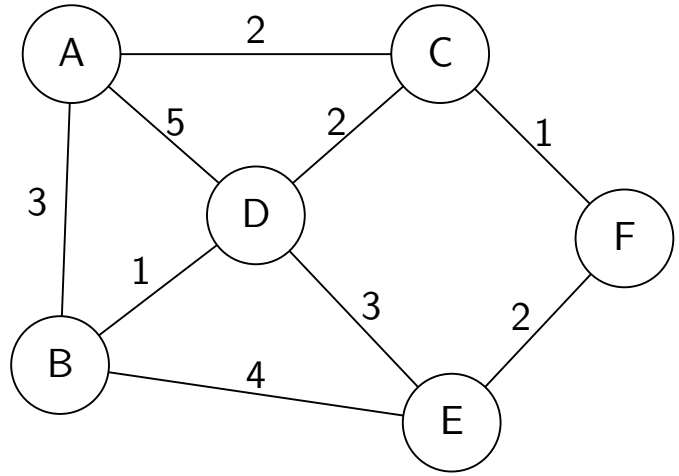
- Cada cierto tiempo todos los routers vecinos intercambian los vectores con distancias a todos los demás routers del AS y se recalculan los vectores de distancias.
- Lo anterior también ocurre cuando un router detecta una variación en la distancia a algunos de sus vecinos.

Ejemplo

- Dada la red de la figura

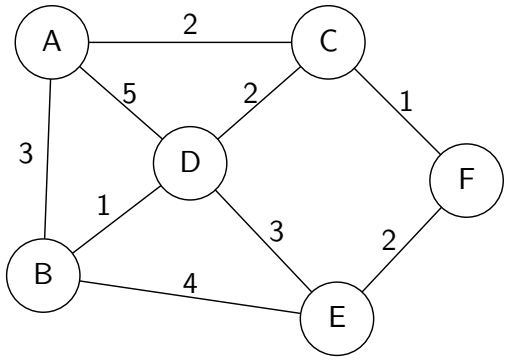


obtener las tablas de encaminamiento según el algoritmo Distance-Vector.



Tablas de encaminamiento iniciales.

	A	B	C	D	E	F
A	0	3/B	2/C	5/D	∞	∞
B	3/A	0	∞	1/D	4/E	∞
C	2/A	∞	0	2/D	∞	1/F
D	5/A	1/B	2/C	0	3/E	∞
E	∞	4/B	∞	3/D	0	2/F
F	∞	∞	1/C	∞	2/E	0

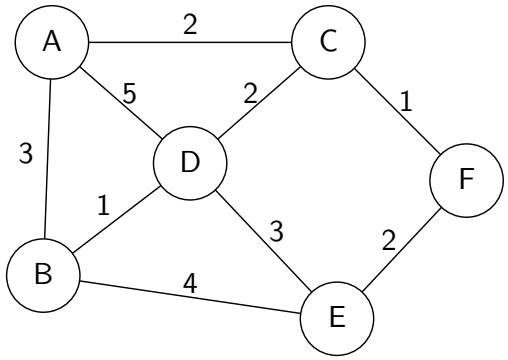


Tablas de encaminamiento iniciales.

	A	B	C	D	E	F
A	0	3/B	2/C	5/D	∞	∞
B	3/A	0	∞	1/D	4/E	∞
C	2/A	∞	0	2/D	∞	1/F
D	5/A	1/B	2/C	0	3/E	∞
E	∞	4/B	∞	3/D	0	2/F
F	∞	∞	1/C	∞	2/E	0

Tablas de encaminamiento
tras el primer intercambio de vectores.

	A	B	C	D	E	F
A	0	3/B	2/C	**4/C**	**7/B**	**3/C**
B	3/A	0	**3/D**	1/D	4/E	**6/E**
C	2/A	**3/D**	0	2/D	**3/F**	1/F
D	**4/B**	1/B	2/C	0	3/E	**3/C**
E	**7/B**	4/B	**3/F**	3/D	0	2/F
F	**3/C**	**6/E**	1/C	**3/C**	2/E	0



Tablas de encaminamiento
tras el primer intercambio de vectores.

	A	B	C	D	E	F
A	0	3/B	2/C	**4/C**	**7/B**	**3/C**
B	3/A	0	**3/D**	1/D	4/E	**6/E**
C	2/A	**3/D**	0	2/D	**3/F**	1/F
D	**4/B**	1/B	2/C	0	3/E	**3/C**
E	**7/B**	4/B	**3/F**	3/D	0	2/F
F	**3/C**	**6/E**	1/C	**3/C**	2/E	0

Tablas de encaminamiento
tras el segundo y definitivo intercambio de vectores.

	A	B	C	D	E	F
A	0	3/B	2/C	4/C	**5/C**	3/C
B	3/A	0	3/D	1/D	4/E	**4/D**
C	2/A	3/D	0	2/D	3/F	1/F
D	4/B	1/B	2/C	0	3/E	3/C
E	**5/F**	4/B	3/F	3/D	0	2/F
F	3/C	**4/C**	1/C	3/C	2/E	0

Capítulo 21

Multicasting (Multidifusión)

21.1. Modelos de transmisión

1. El **modelo unicast** donde un emisor realiza un envío y los datos llegan a un único receptor. Ejemplo: las transmisiones TCP.

2. El **modelo multicast**: Un emisor realiza un envío y los datos llegan a muchos receptores. Ejemplos: actualizaciones de software, streaming de audio y vídeo, pizarras compartidas, juegos interactivos, ... A nivel de la capa de red, el multicasting permite ahorrar ancho de banda (sobre todo en en el enlace de subida el emisor) si lo comparamos con la versión unicast de las aplicaciones.

    - El **broadcasting** se considera un caso particular del modelo multicast en el que los datos llegan a todos los posibles receptores.

21.2. El multicasting a nivel de aplicación y de red

- El multicasting puede realizarse fundamentalmente de dos maneras:

  1. A **nivel de la capa de aplicación**, mediante dos técnicas diferentes:

     a) **Múltiple-unicast:** el emisor genera una copia de los datos para cada posible receptor.

     b) **Mediante overlay networks:** en este caso, el emisor sólo genera una copia y en algunos puntos estratégicos de la red se coloca un programa de replicación que hace las veces de router multicast.

  2. A **nivel de la capa de red**, emitiendo una única copia de los datos. Los routers multicast se encargan de replicar los paquetes que transportan los datos tanto como sea necesario.

- Las dos últimas alternativas tienen dos grandes ventajas:
  1. La redundancia de la emisión es mucho menor.
  2. El emisor no tiene por qué conocer los posibles receptores (que pueden ser muchos).
- La tercera además simplifica el software a nivel de aplicación.

21.3. Algoritmos de broadcasting

21.3.1. Flooding

- Los routers envían los paquetes recibidos a todos sus vecinos, excepto al que se los entrega.
- Para evitar que los lazos generen una sucesión infinita de replicaciones (*broadcast storm*) se utilizan dos técnias:
  1. **Sequence-number-controlled flooding**. En cada paquete se incluye un número de secuencia que lo identifica. Este número junto con la dir IP origen del paquete sirve para que cuando un router lo recibe, éste lo replica si y sólo si antes no ha sido replicado.
  2. **Reverse path forwarding (RPF)**. En este caso se utiliza sólo la dir IP del paquete. Un router lo replica si y sólo si recibe el paquete a través del enlace que forma parte de la ruta de distancia mínima entre ambos routers (el de origen del paquete

y el que recibe el paquete).[1] Un ejemplo lo encontramos en la siguiente figura (A es el router origen):

[1] Nótese que esta información figura en la tabla de routing de cada router que recibe el paquete.

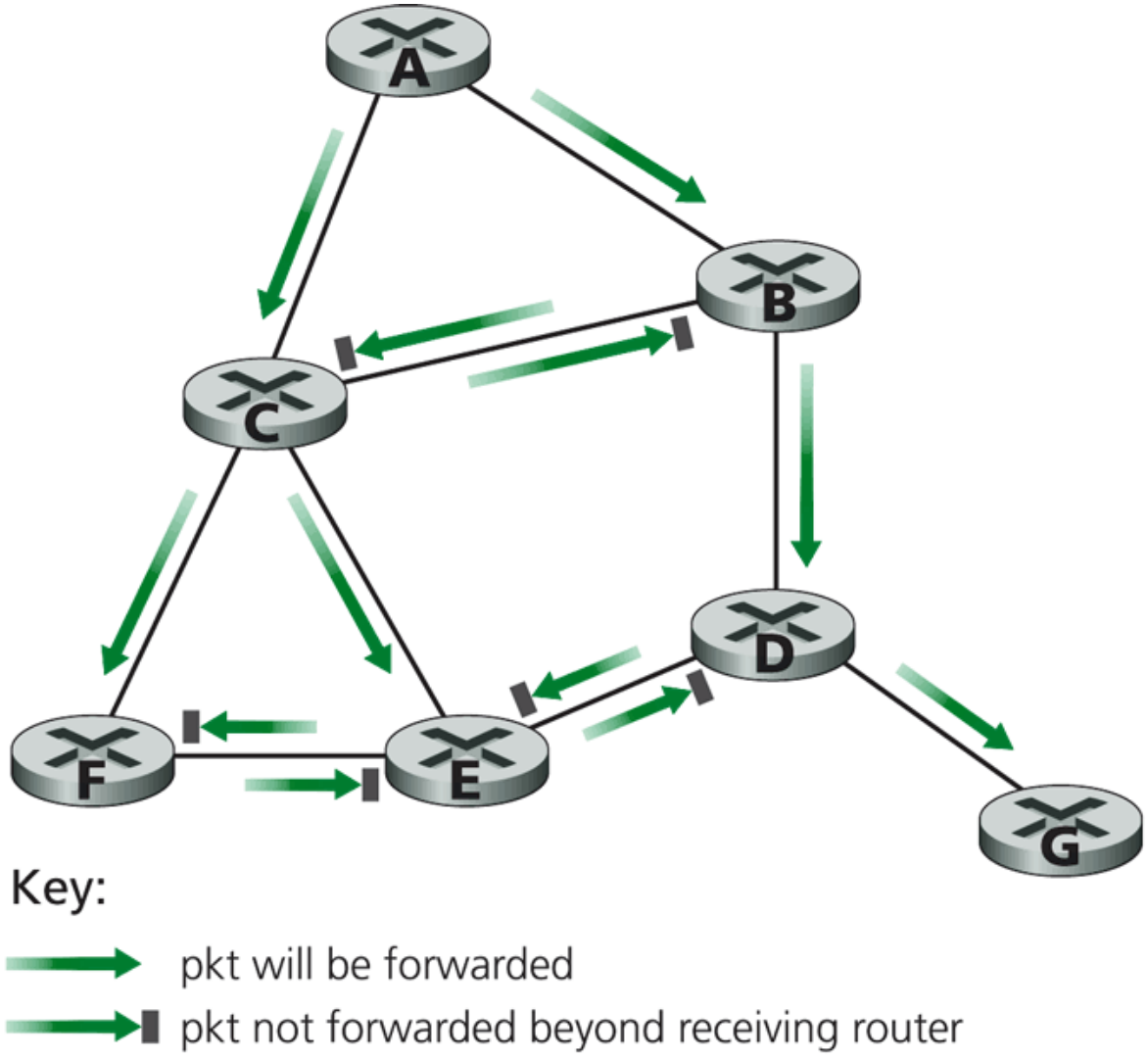


Figure 4.41 ♦ Reverse path forwarding

21.3.2. Spanning-tree broadcast

- Es más óptimo que el fooding (cada router recibe sólo una copia de los datos).
- Se basa en que los routers sólo envían los datos a través del *minimum spanning tree*. El árbol de expansión mínimo es un árbol (y por tanto un grafo sin ciclos) que utiliza a todos los nodos del grafo y que posee un coste mínimo.

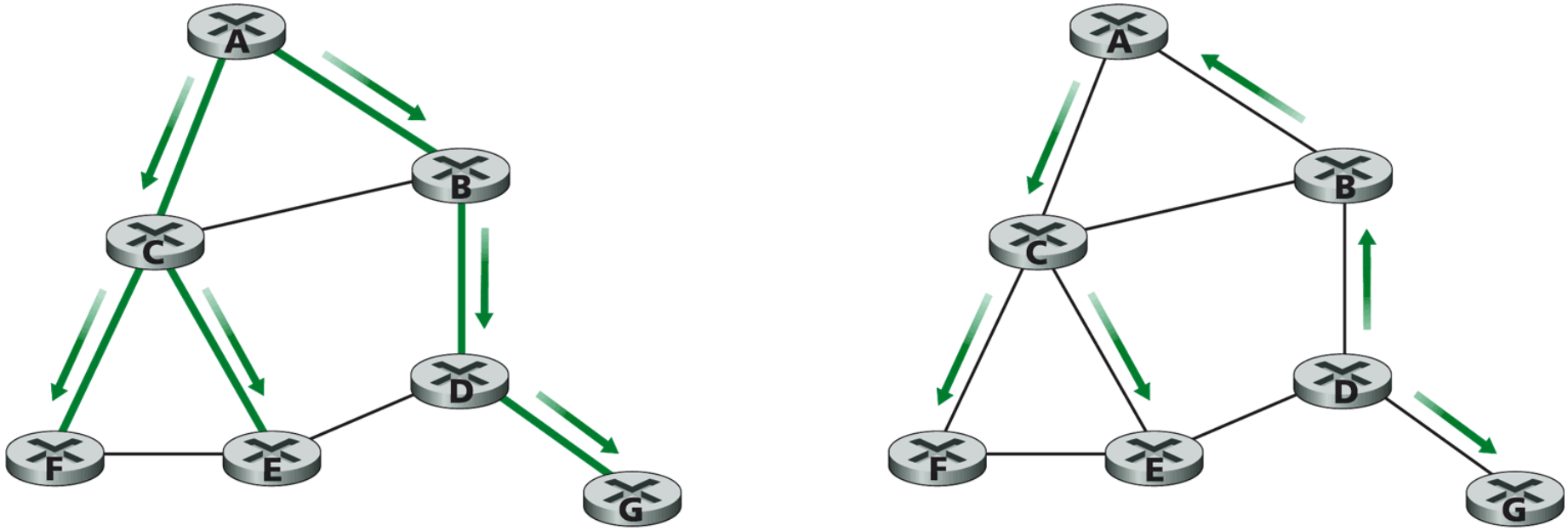


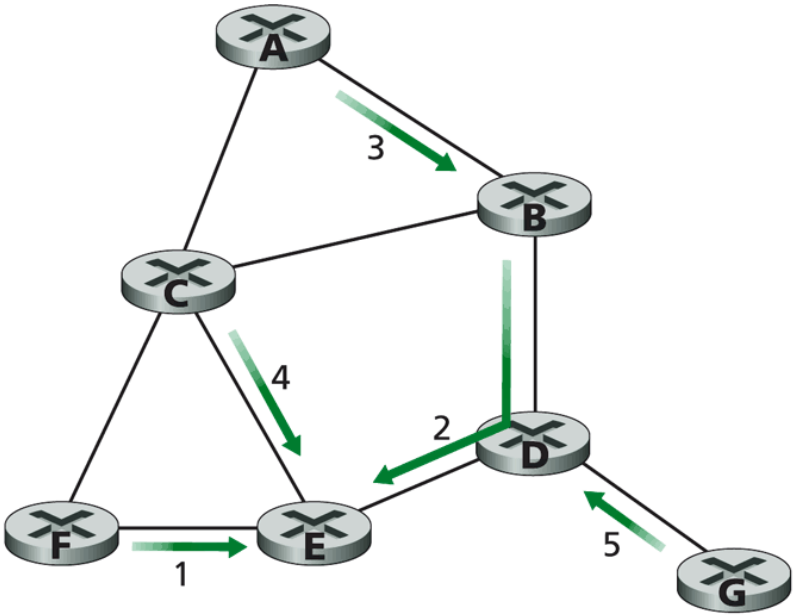
Figure 4.42 ♦ Broadcast along a spanning tree

- Para determinar el *minimum spanning tree* se puede utilizar el **algoritmo de Steiner** [14], pero éste es muy costoso en términos de tiempo (es un problema de complejidad exponencial $\mathcal{O}(x^N)$, donde N es el número de nodos del grafo y x es una costante) y por este motivo se utilizan otros algoritmos menos costosos basados en heurísticas.

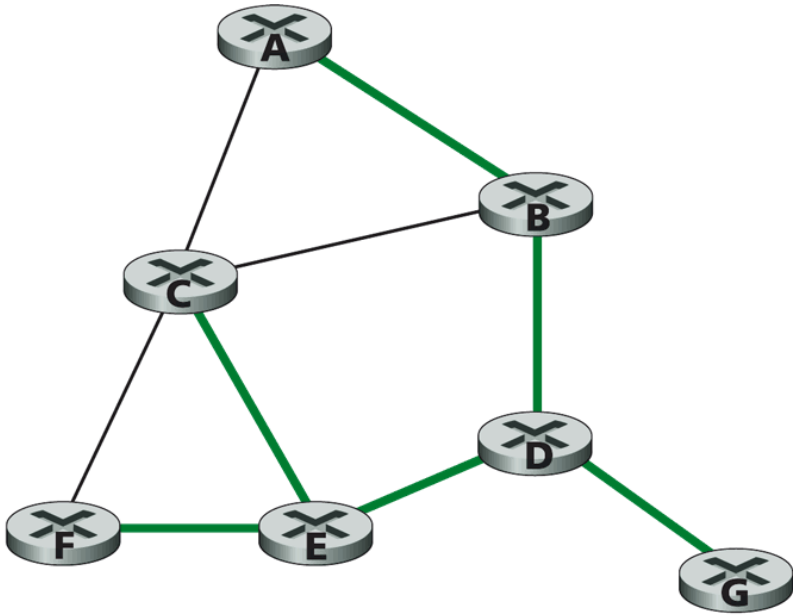
- Uno de los algoritmos usados es el **"algoritmo del punto de encuentro (rendezvous point) para el cálculo del spanning tree"**[2]:

  1. Definir un *rendezvous router* y hacérselo saber a todos los demás routers.
  2. Todos los demás routers envían un mensaje *tree-join* hacia el *rendezvous router* usando el camino mínimo (que sí es determinado por los algoritmos de routing unicast).
  3. Cada router que recibe un mensaje *tree-join* añade al *spanning tree* el enlace por el que ha llegado el mensaje.
  4. El mensaje se retransmite hacia el *rendezvous router* si el router todavía no pertenece al *spanning tree*. Ejemplo (E es el punto de encuentro):

[2] En ocasiones llamado también "aproximación basada en el centro para el cálculo del spanning tree".



a. Stepwise construction of spanning tree



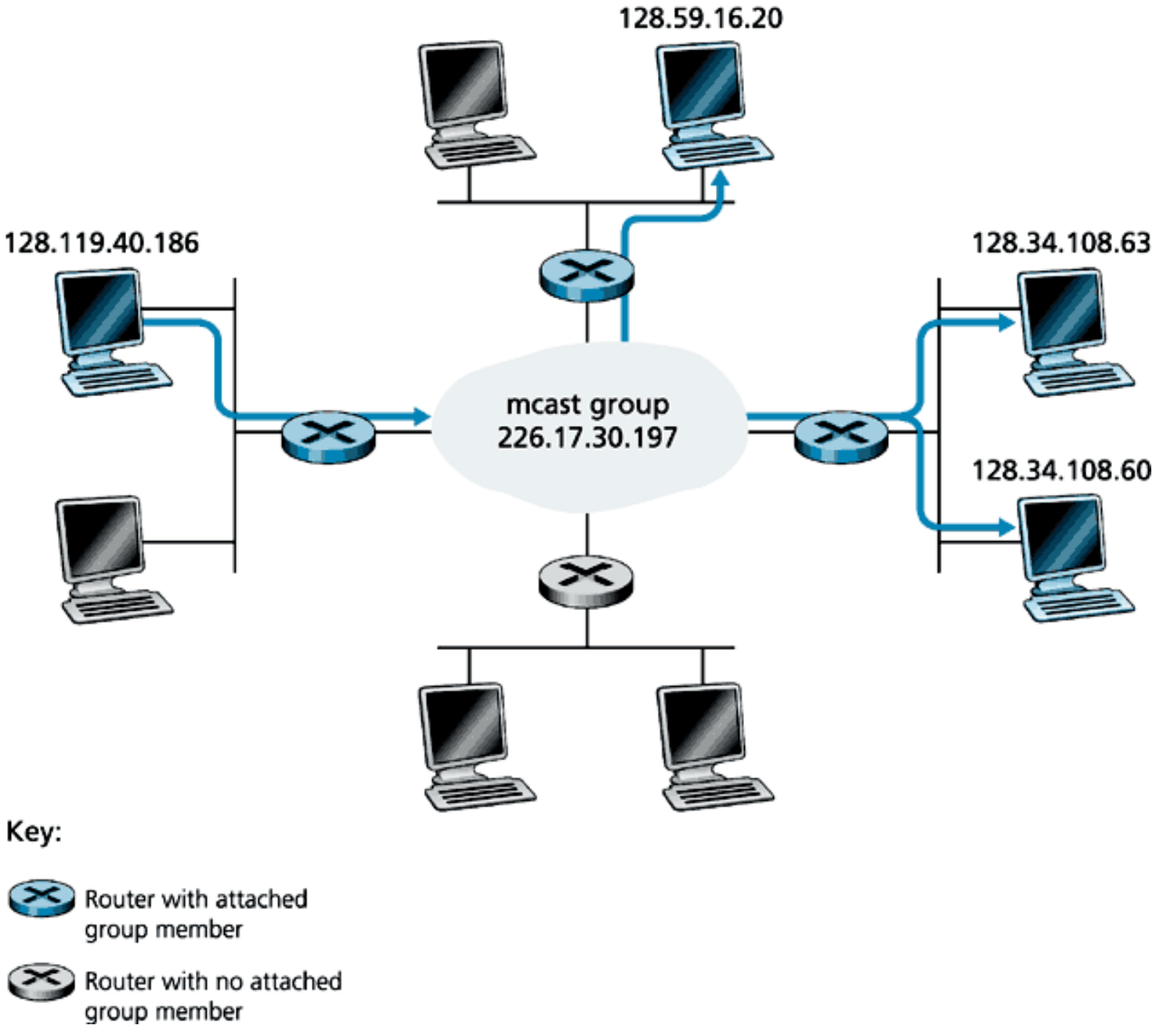
b. Constructed spanning tree

Figure 4.43 ♦ Center-based construction of a spanning tree

21.4. Los grupos multicast

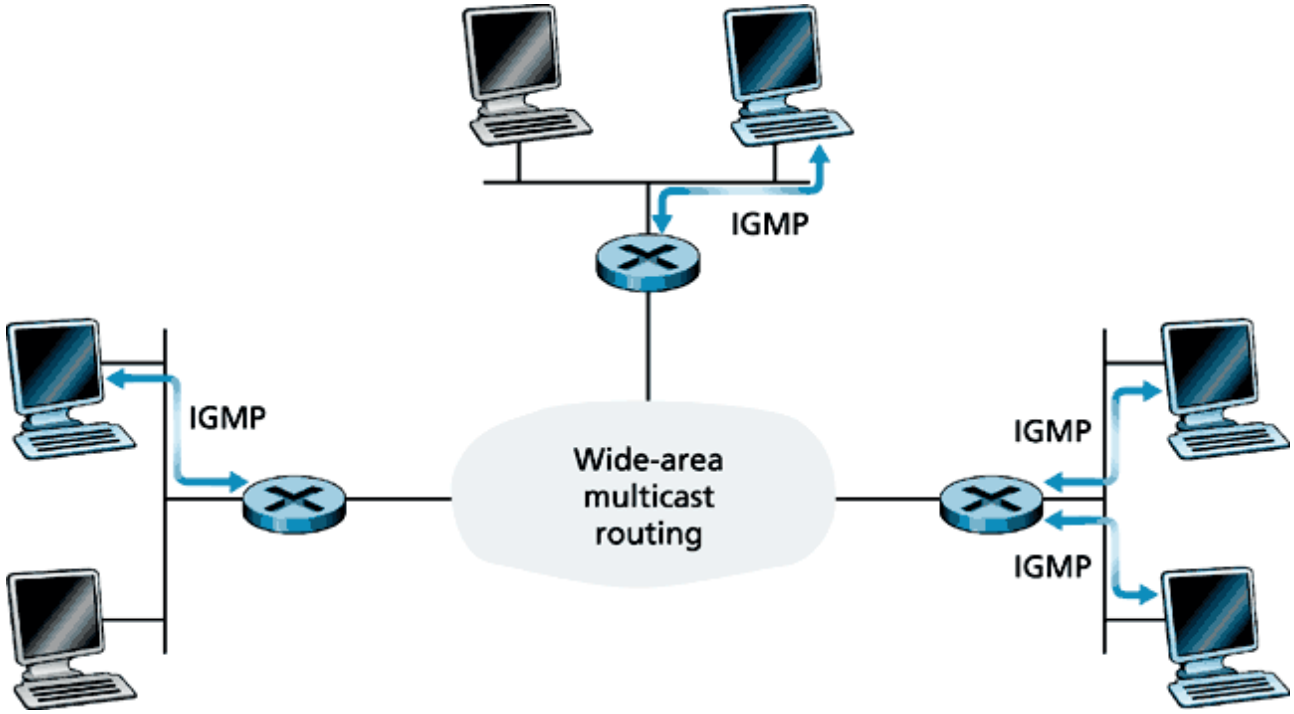
- La clase D de dirs IP fueron reservadas para grupos multicast. La red multicast es la 224.0.0.0/4 [14]. Véase la transparencia 18.2.
- Todos los hosts que escuchan una dir IP multicast forman un **grupo multicast**. Existen 2^{32-4} grupos multicast diferentes.
- Cada dir IP multicast se comporta como un canal de tipo broadcast (como los canales de radio) donde todos los hosts pertenecientes a un grupo multicast puede enviar y recibir datagramas.
- El envío de datagramas a un grupo multicast puede hacerse en cualquier instante, independientemende de que en ese momento exista otro emisor en el grupo.
- En un datagrama dirigido a un grupo multicast figura, como dir IP destino, la dir IP del grupo multicast.
- Los miembros de un grupo multicast no conocen (al menos por la capa de red) a los otros miembros del grupo.

- Nadie controla quién pertenece a un grupo multicast (a nivel de la capa de red). La limitación la imponen los routers multicast (filtrando) y los TTL's de los paquetes.
- Nadie controla qué grupos multicast se crean o destruyen (a nivel de red). A nivel de aplicación sí que se hace (véase la aplicación *sdr* (Session DiRectory)) diseñada para el MBone [12].
- Cuando un host escribe a una dir multicast, el datagrama llega a todos los miembros del grupo. En el siguiente ejemplo se supone que el grupo multicast 226.17.30.197 está formado por los 3 receptores 128.59.16.20, 128.34.108.63 y 128.34.108.60, y por el emisor 128.119.40.186.

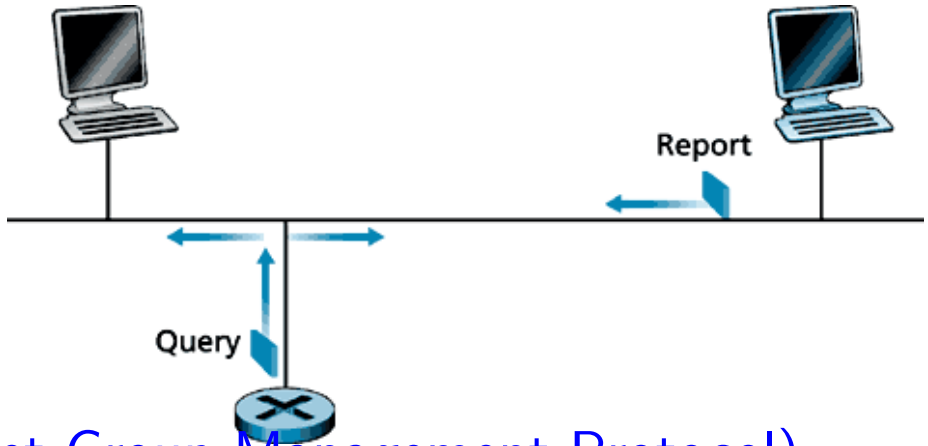

128.59.16.20
128.119.40.186
128.34.108.63
mcast group
226.17.30.197
128.34.108.60
Key:
Router with attached group member
Router with no attached group member

21.5. El IGMP (Internet Group Management Protocol)

- RFC 2236.
- Controla cómo se crean y destruyen los grupos multicast.
- El IGMP se utiliza entre un host y su router multicast, que por definición debe de estar en su red.



- Funcionamiento:



1. El router multicast envía periódicamente un mensaje del tipo `membership_query` a la(s) red(es) conectada(s) para preguntar si existe al menos un host suscrito a un grupo multicast.
2. Todos los hosts suscritos a algún grupo multicast contestan (esperando un tiempo aleatorio máximo establecido en el mensaje `membership_query`) con mensajes del tipo `membership_report`, indicando en cada uno de ellos las dirs IP multicast correspondientes. Esperan un tiempo aleatorio antes de contestar porque si durante la espera escuchan un `membership_report` de otro host de la sub-red indicando que está suscrito a ese grupo multicast, se ahorran la contestación.

- Si el router no recibe ningún `membership_report` en un cierto tiempo después de un `membership_query`, deja de participar en las transmisiones multicast (pruning del árbol multicast).
- El anterior estado también puede conseguirse enviando al router un mensaje opcional llamado `leave_group`.

21.6. Protocolos de routing multicast

21.6.1. El DVMRP (Distance Vector Multicast Routing Protocol)

- RFC 1075.
- Utiliza el algoritmo de flooding RPF con poda (pruning).
- Los mensajes de poda deshabilitan las transmisiones multicast durante 2 horas y sirven para que los routers que no quieren recibir el tráfico multicast dejen de recibirlo [15].

21.6.2. PIM (Protocol Independent Multicast)

- RFC's 2362, 2201 y 2189.
- Posee dos modos de funcionamiento:
  1. **Modo de funcionamiento denso:**
     - Se emplea cuando la mayoría de los routers multicast están involucrados en la transmisión multicast.
     - Utiliza el algoritmo de flooding RPF con poda.
  2. **Modo de funcionamiento disperso:**
     - Se emplea cuando sólo una minoría de los routers multicast están involucrados en la transmisión multicast.
     - Utiliza el algoritmo spanning-tree broadcasting.

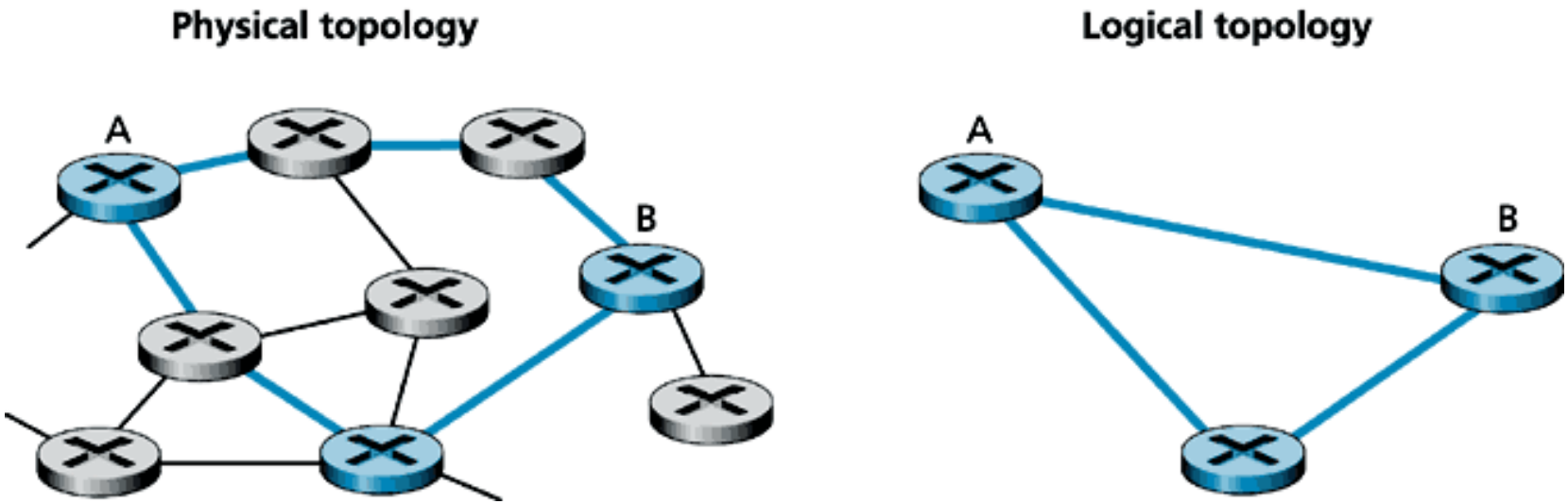
21.7. Multicasting en las redes locales

- Todas las tecnologías de red que actualmente existen (independientemente de la existencia de un router multicast) soportan que un host envíe un datagrama a todos de hosts de la sub-red (broadcasting).
- Esto se puede hacer fácilmente si enviamos el datagrama a la dir de broadcast de la sub-red (la última del rango de direcciones).
- Los routers (normalmente) ignoran los datagramas dirigidos a la dir de broadcast de la sub-red.

21.8. Multicasting en Internet: el MBone (Multicast BackbONE)

- ¡Todos los routers de Internet no tienen que ser multicast!
- Debido a los grandes requerimientos de ancho de banda que serían necesarios, en general no está permitido que ningún datagrama multicast acceda a Internet y alcance potencialmente a todos sus nodos, a no ser que sea transmitido a través de redes virtuales especialmente diseñadas con ese propósito como el MBone [26, 1].
- El MBone es una red virtual, formada por routers multicast conectados entre sí mediante tunneling (para distribuir tráfico multicast sobre los routers unicast de Internet).

Ejemplo:



El router A encapsula los datagramas multicast en datagramas IP unicast con origen A y con destino (B, ...). B los des-encapsula, ve el datagrama multicast con el destino original y realiza de nuevo tunneling si fuera necesario.

- Evidéntemente, para que un host pueda enviar o recibir datos desde el MBone, tiene que existir un router multicast perteneciente al MBone en su sub-red.

- La distancia (en hops) que recorren los datagramas multicast en el MBone depende:
  1. Del TTL asignado por el host emisor.
  2. De que los routers permitan su retranmisión.
- RedIRIS (la red de investigación española) pertenece actualmente al MBone [13, 26].
- The Roling Stones fueron los primeros en utilizar comercialmente el MBone en 1994. Emitieron 25 minutos de un concierto dado en Dallas, USA [20].
- El MBone dejará de existir como red virtual cuando todos los routers de Internet sean multicast.

Capítulo 22

Mobility (Movilidad)

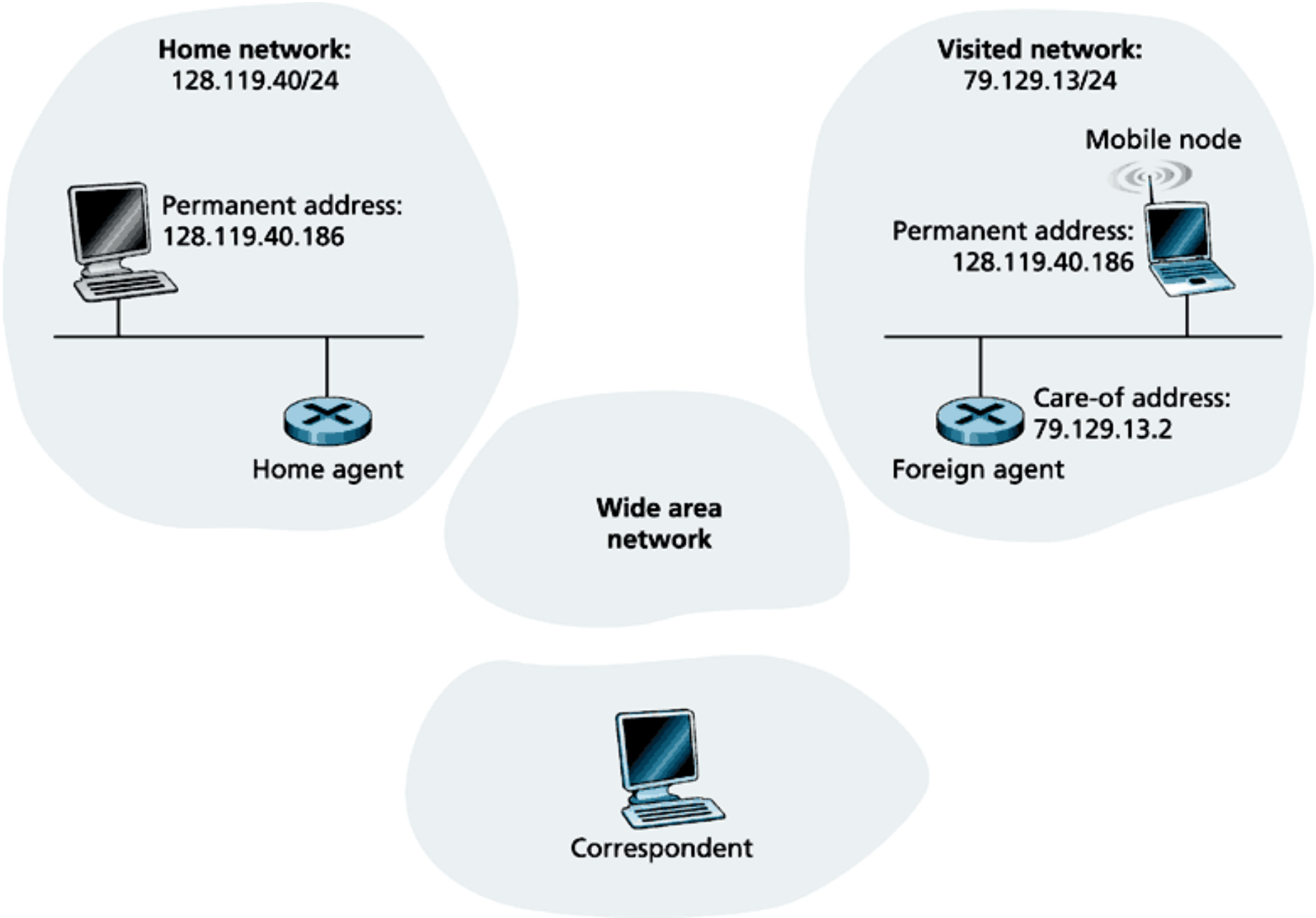
22.1. Routing para hosts móviles

- RFC 3220 (Mobile IP).
- El IP contempla la posibilidad de que uno o dos de los hosts que han establecido una conexión se puedan mover (**cambiar de red física**) a lo largo del tiempo que dura la conexión [14]. Esto puede ocurrir si se utilizan redes wireless.
- Siempre que el host móvil desee recibir datos (de una determinada conexión) debe conservar su dir IP:
  - Usando UDP, si el host que es móvil envía pero nunca recibe datos, entonces no es necesario añadir ninguna funcionalidad a la capa de red. Sin embargo, si el host que es móvil recibe datos entonces el routing estático no funciona.
  - Usando TCP, como la conexión es siempre duplex (se envían datos de alguna clase en ambos sentidos), la capa de red debe de permitir el routing para host móviles.

- El routing para host móviles puede ser utilizado para desviar paquetes hacia un host "malicioso". Por este motivo todos los protocolos utilizados poseen sistemas de identificación.

22.2. Nomenclatura

- **Mobile host:** el host que se mueve entre redes durante una conexión.
- **Correspondent host:** el host (móvil o no) que mantiene la conexión con el host móvil.
- **Correspondent agent:** el gateway del correspondent host.
- **Home network:** la red originaria del host móvil.
- **Foreing network:** la red a la que se mueve el host móvil con una conexión establecida.
- **Home agent:** un router perteneciente a la home network que conoce la dir IP del host móvil y la red foreing network.
- **Foreing agent:** un router perteneciente a la foreing network que conoce la dir IP (fija) del host móvil.
- **COA (Care-of Address):** dir IP del *foreing agent*.


Home network:
128.119.40/24
Permanent address:
128.119.40.186
Home agent
Wide area
network
Visited network:
79.129.13/24
Mobile node
Permanent address:
128.119.40.186
Care-of address:
79.129.13.2
Foreign agent
Correspondent

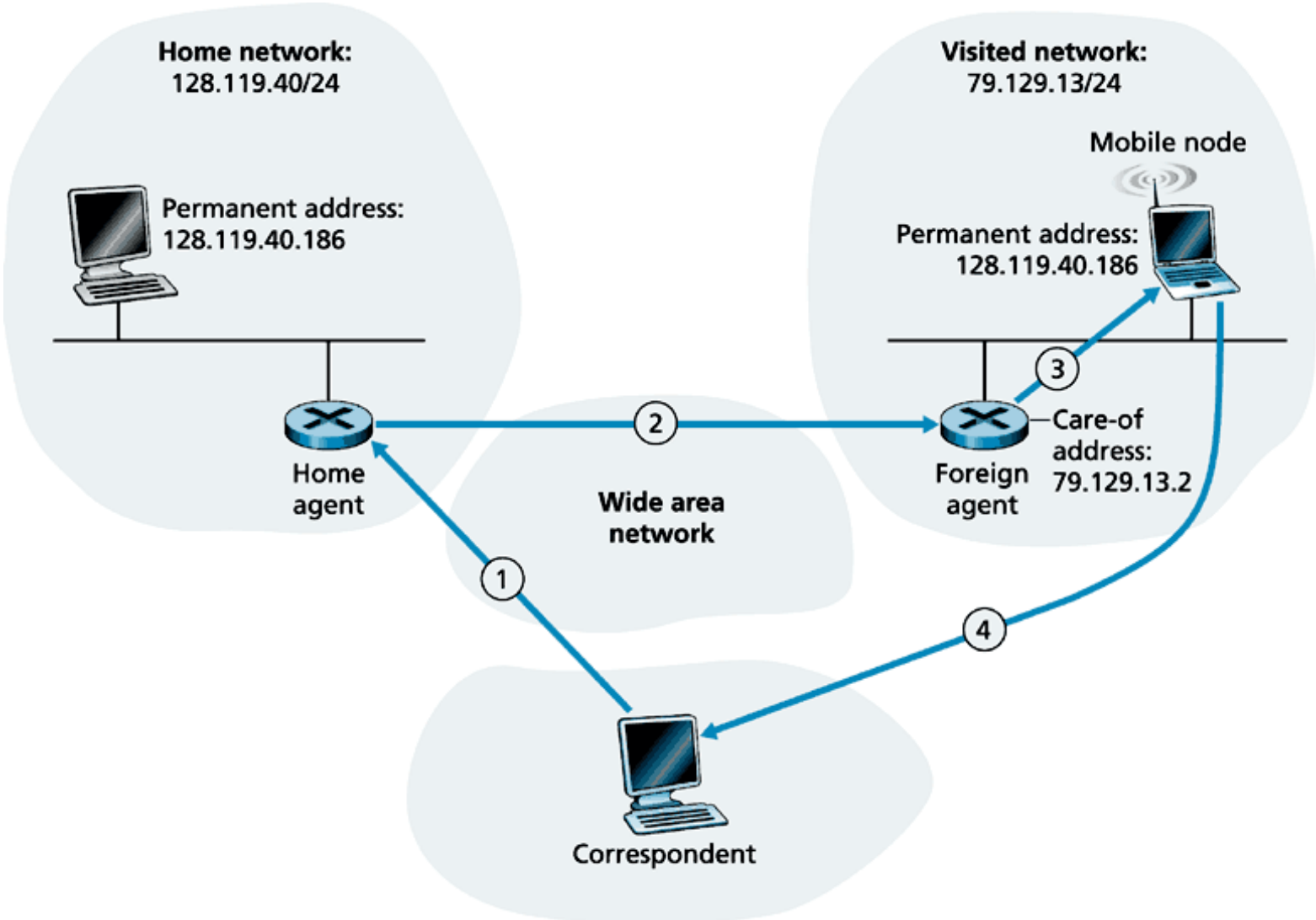
22.3. Routing indirecto

- Consiste en los siguientes pasos:

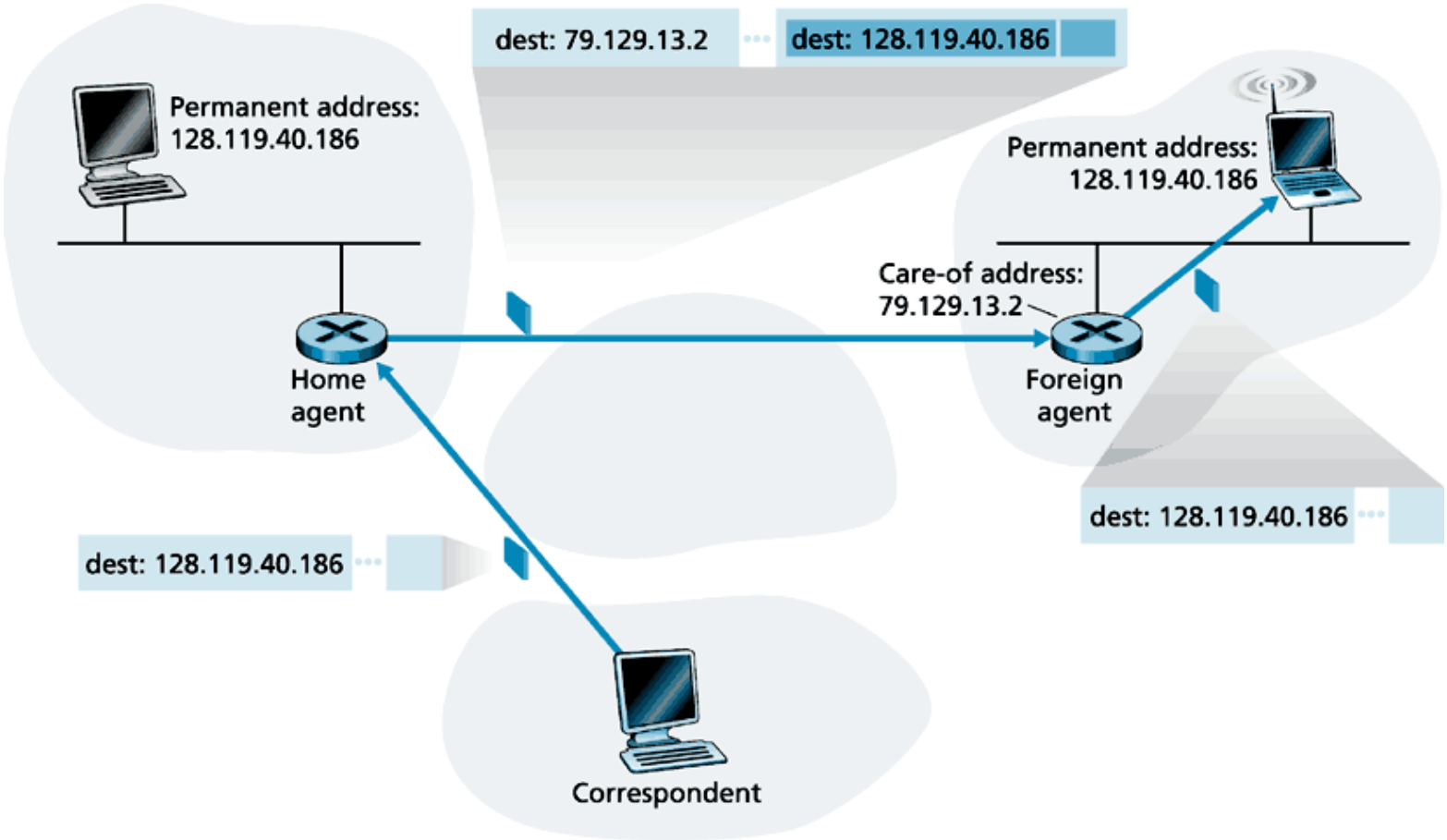
  1. El *correspondent host* envía los paquetes ageno a la situación geográfica del *mobile host* utilizando su dir IP permanente (la que usa en su *home network*).
  2. Los paquetes llegan hasta el *home agent* y éste los envía al *foreing agent* usando la COA. Los paquetes originales se encapsulan (mediante tunneling, RFC's 2003 y 2004) en otros para conservar las dirs IP del *correspondent host* y del *mobile host*. Esto es necesario porque ningún router intermedio entre el *home agent* y el *foreing agent* encaminaría adecuadamente los paquetes encapsulados.
  3. Los paquetes encapsulados llegan hasta el *foreing agent*, los desencapsula y los envía al *mobile host*.
  4. El *mobile host* puede contestar directamente al *correspondent host* porque éste no es móvil.

Ejemplo

- Encaminamiento:



- Tunneling:

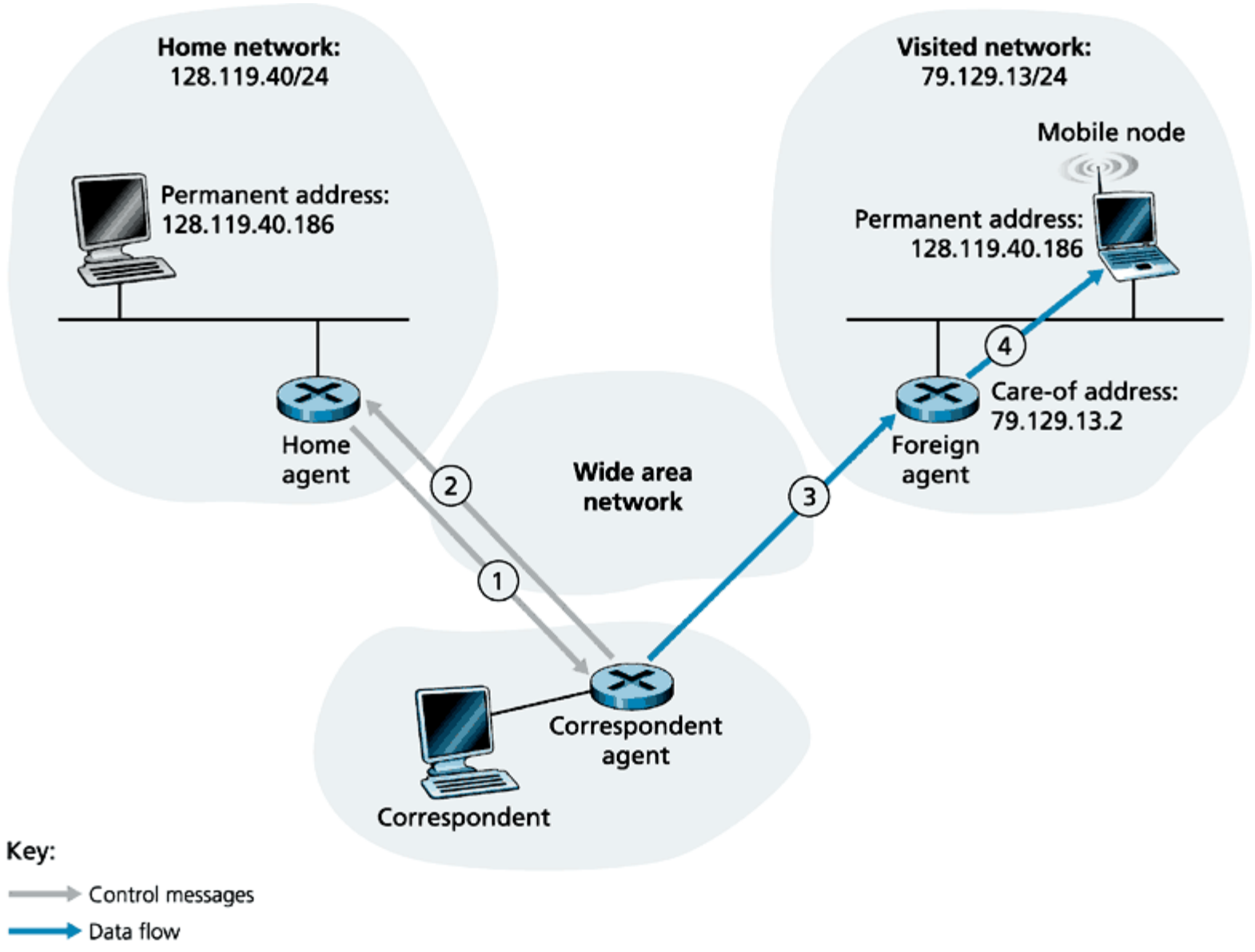

dest: 79.129.13.2
dest: 128.119.40.186
Permanent address:
128.119.40.186
Permanent address:
128.119.40.186
Care-of address:
79.129.13.2
Home
agent
Foreign
agent
dest: 128.119.40.186
dest: 128.119.40.186
Correspondent

22.4. Routing directo

- Intenta reducir la distancia recorrida por los paquetes enviados al *mobile host*.[1] En este caso el *correspondent agent* no es ageno a que el *mobile host* es realmente móvil.
- El *correspondent agent* conoce que el *mobile host* está en la *foreing network* (de hecho conoce la COA) porque se lo indica el *home agent*. Esto ocurre cuando el *home agent* ve que hay tráfico para el *mobile host* que ya no puede entregar.
- El *correspondent agent* envía los paquetes al *mobile host* usando tunneling. Estos llegan encapsulados al *foreing agent*. El *foreing agent* desencapsula los paquetes y los entrega al *mobile host*.

[1] Imagínese que el host móvil está en la misma red que el *correspondent host*.

Ejemplo


Home network:
128.119.40/24
Permanent address:
128.119.40.186
Home
agent
Wide area
network
Visited network:
79.129.13/24
Mobile node
Permanent address:
128.119.40.186
Care-of address:
79.129.13.2
Foreign
agent
Correspondent
agent
Correspondent
1
2
3
4
Key:
Control messages
Data flow

Parte V

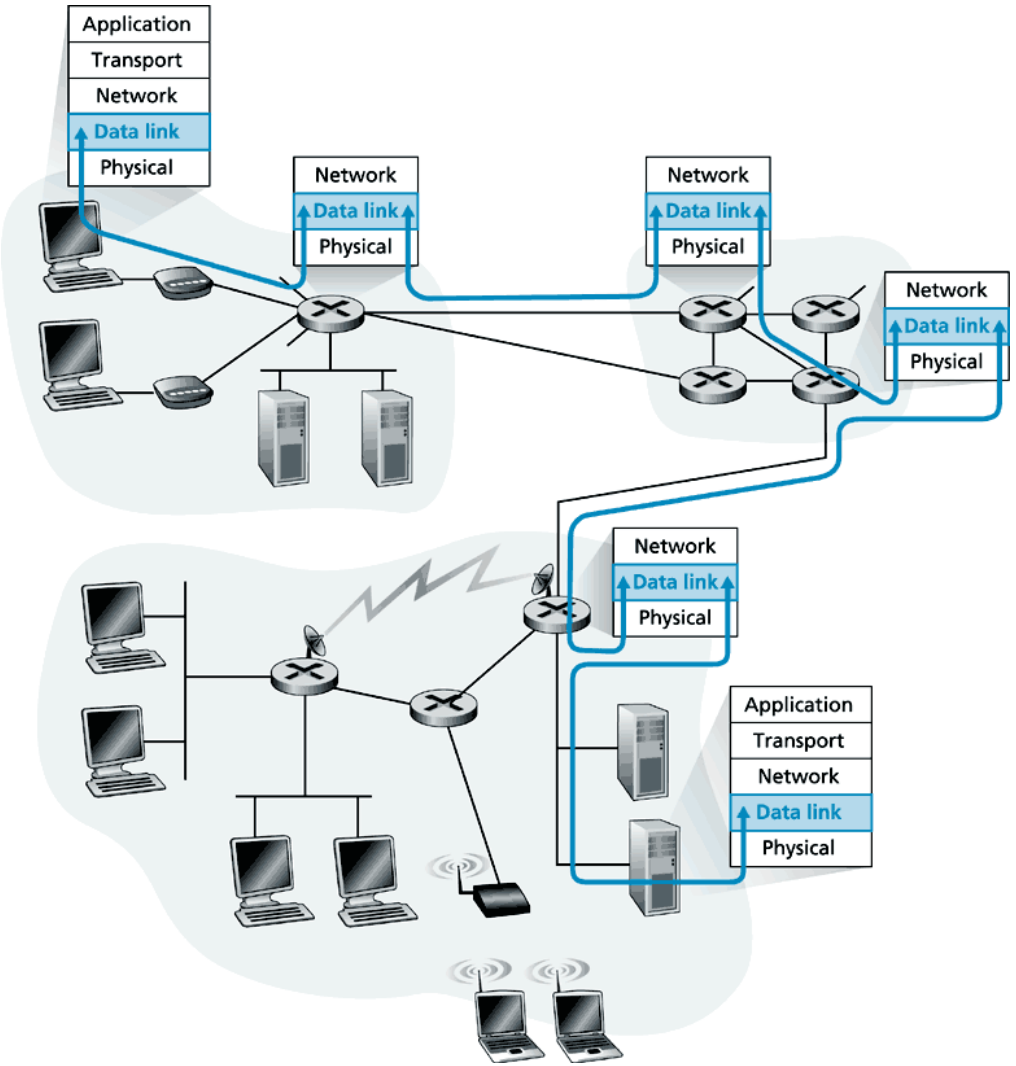
La capa de enlace de datos

Capítulo 23

Servicios de la capa de enlace de datos

23.1. El marco de trabajo

- La capa de enlace de datos se sitúa entre la capa de red y la capa física.



- La capa física está generalmente muy relacionada con la capa de enlace de datos (se dice que ambas son muy dependientes). Sin embargo, en algunas tecnologías (como ATM) existe una diferenciación clara.
- La transmisión de un paquete desde el host emisor al receptor implica generalmente el uso de diferentes tecnologías con distintas capas de enlace de datos, cada una con un modelo de servicio específico.

23.2. Nodos, frames y enlaces

- La capa de enlace de datos transmite bloques de datos (a los que llamaremos *frames* [14]) entre dos nodos (hosts o routers) de la red adyacentes, a través de un enlace de transmisión.
- En el contexto de Internet, cada frame puede transmitir un paquete (o una parte de éste) generado por la capa de red.
- A nivel de enlace de datos no tiene sentido distinguir entre hosts y routers, y por eso en adelante emplearemos sólo el término **nodo**. El problema de transmitir un frame es el mismo, independientemente del tipo de nodo.

23.3. Servicios generalmente proporcionados

Transporte de datos

- El único servicio que siempre se suministra independientemente de la tecnología subyacente, es el de transporte de datos. Los demás son opcionales (para la capa de red y de transporte).

Direccionamiento

- En el caso de que en un mismo enlace existan más de un posible nodo receptor, la capa de enlace de datos proporciona un mecanismo de direccionamiento basado en direcciones que permita al emisor diferenciarlos.
- LLamaremos a estas direcciones, **direcciones físicas**.

Control de acceso al medio

- Media Access Control (MAC).
- En aquellos casos donde existan dos o más nodos emisores potenciales conectados a un mismo enlace de datos, la capa de enlace de datos proporciona un mecanismo de arbitraje del medio para evitar las **colisiones**.
- Por definición, se produce una colisión cuando dos o más emisores acceden a un medio compartido para enviar datos y lo hacen al mismo tiempo (cuando esto el medio no lo permite).

Transferencia fiable

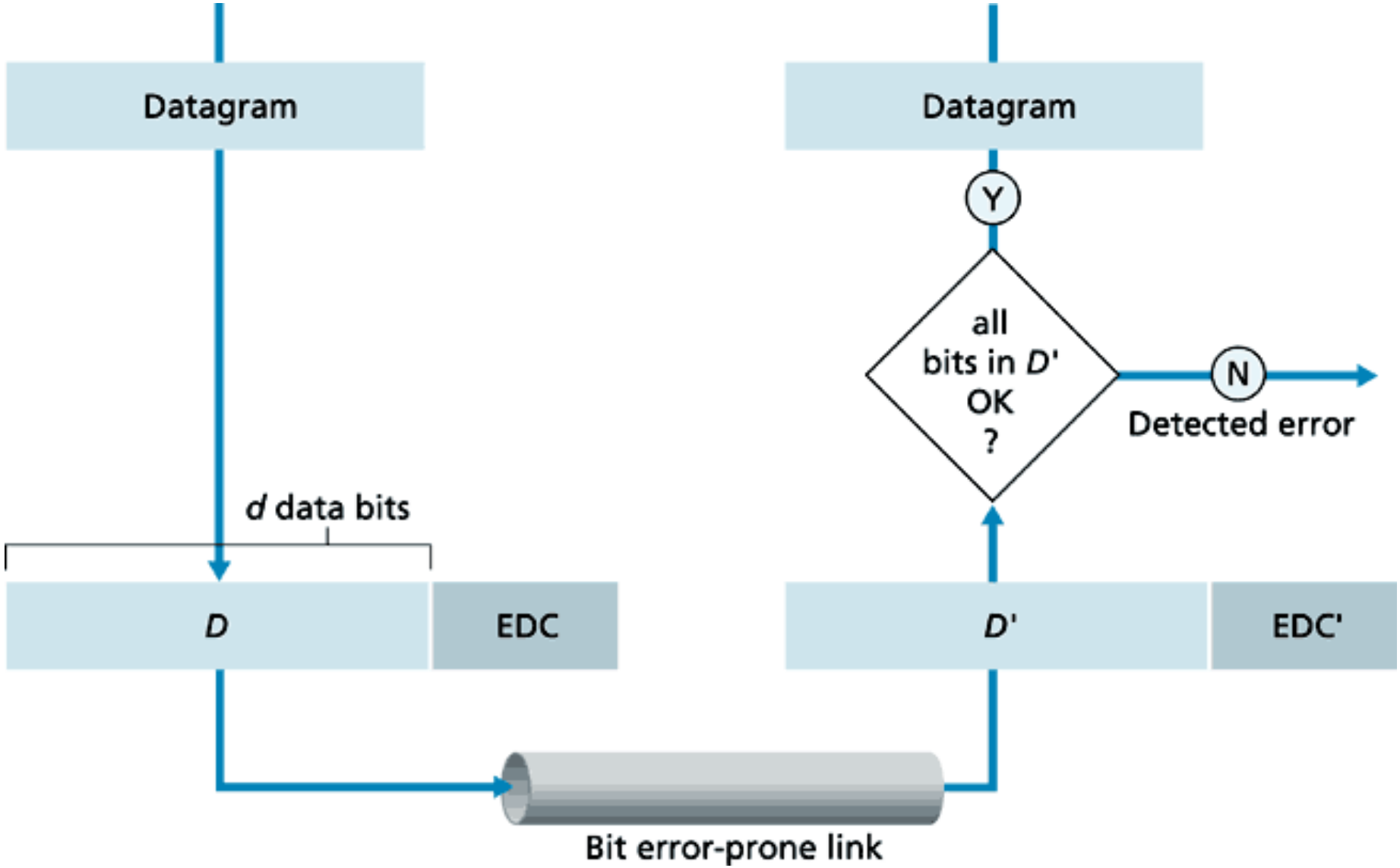
- En algunas tecnologías, la capa de enlace de datos garantiza que el transporte de un frame entre el emisor y el receptor se realiza sin errores. Para ello se implementan, a este nivel, algoritmos de control de flujo y de errores.
- Aunque puede pensarse que este servicio se solapa con el que se presta en la capa de transporte, debe tenerse en cuenta que a este nivel los errores sólo se corrigen si se producen sobre el enlace (no en un router por ejemplo).
- Además, en aquellos casos donde la probabilidad de error del enlace es bastante alta (como puede ocurrir en los enlaces de larga distancia o en los enlaces de radio), es mucho más eficiente controlar los errores nodo-a-nodo que de extremo-a-extremo (como hace el TCP).

Capítulo 24

Control de errores

24.1. Fundamentos

- Las **técnicas de detección de errores** sólo pueden detectar (con una cierta probabilidad de acierto) si existen errores de transmisión y éstos sólo pueden corregirse usando retransmisión de datos. Las **técnicas de corrección de errores** además permiten corregirlos sin retransmisión.
- En ambos casos se introduce **información redundante**, especialmente en el segundo (por eso su uso no es muy frecuente en redes de transmisión de datos a no ser que los errores de transmisión sean muy frecuentes, las latencias muy grandes o las comunicaciones simplex, donde el control de errores mediante ARQ no es posible).
- La información de control de errores es añadida por la capa de enlace de datos del emisor y eliminada por la capa de enlace de datos del receptor:

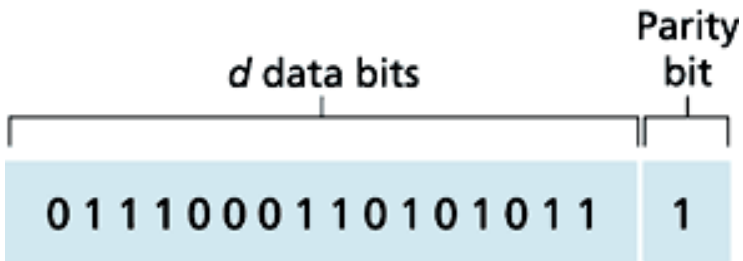


EDC (Error Detecting Code).

24.2. Paridad

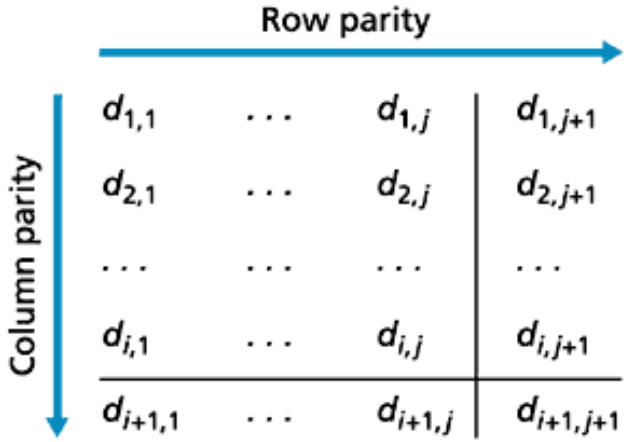
- Consiste en añadir bits de paridad de forma que el número de bits iguales a 1 en el mensaje sea un número impar si usamos paridad impar o un número par si usamos paridad par [14].
- Cuantos más bits de paridad introducimos más errores podemos detectar. Si estos son suficientes, incluso corregirlos. Ejemplos:

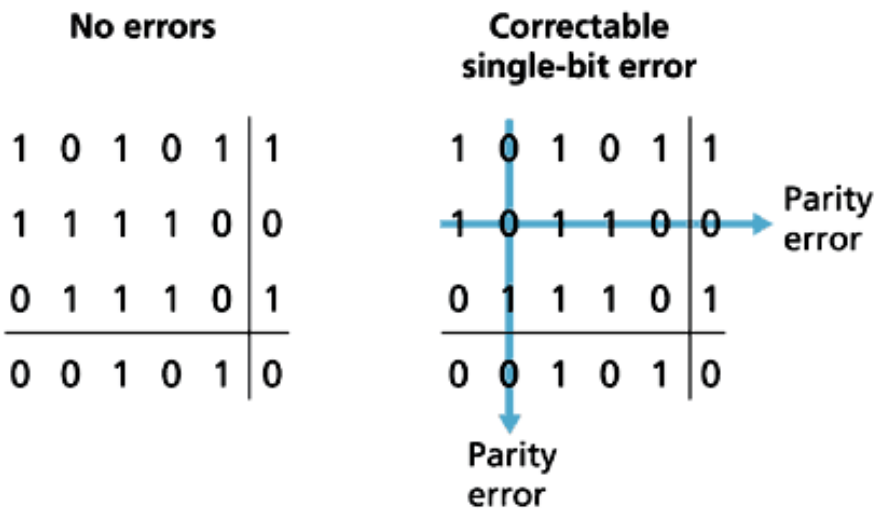
  1. **Paridad simple**. Consiste en añadir un único bit de paridad. En el siguiente ejemplo se utiliza paridad par:



     La probabilidad de detectar (no de corregir) un error es del $50\,\%$.

2. **Paridad bidimensional**. Consiste en añadir bits de paridad por filas y por columnas. En el ejemplo se utiliza paridad par y como puede verse permite corregir un error.





24.3. Checksum

- Consiste en sumar (usando una determinada precisión aritmética) todas las palabras del mensaje y transmitir como EDC dicha suma [22]. El receptor realiza la misma suma y si no concuerda con el EDC se sabe que se ha producido un error de transmisión.
- Como ejemplo se muestra (en C) el algoritmo de la suma de comprobación usado por el TCP/IP:

```
unsigned short cksum (unsigned short *buf, int count) { /*  1 */
  register unsigned long sum=0;                         /*  2 */
  while (count--) {                                     /*  3 */
    sum += *buff++;                                     /*  4 */
    if (sum & 0xFFFF0000) {                             /*  5 */
      sum &= 0xFFFF;                                    /*  6 */
      sum++;                                            /*  7 */
    }                                                   /*  8 */
  }                                                     /*  9 */
  return ~(sum & 0xFFFF);                               /* 10 */
}                                                       /* 11 */
```

- Tanto el emisor como el receptor ejecutan el mismo algoritmo. Lo único que cambia es que el receptor incluye la suma de comprobación en el cálculo (y el emisor no). Si el resultado es 0, entonces se supone que no existen errores de transmisión.

24.4. CRC (Cyclic Redundancy Check)

- Un mensaje de $n+1$ bits puede ser considerado como un polinomio de grado n donde los coeficientes son 0 o 1 [32]. Por ejemplo, el mensaje 00100011 equivaldría al polinomio $x^5 + x + 1$.

- Sea $M_n(x)$ el mensaje (polinomio de grado n) de $n+1$ bits que el emisor quiere enviar al receptor, y sea $G_k(x)$ el polinomio generador usado para crear el CRC. El CRC se calcula realizando

$$R_{k-1}(x) \leftarrow (M_n(x) \times x^k) \text{ mod } G_k(x).$$

- El emisor concatena a $M_n(x)$ los k bits del CRC y los envía. Matemáticamente:

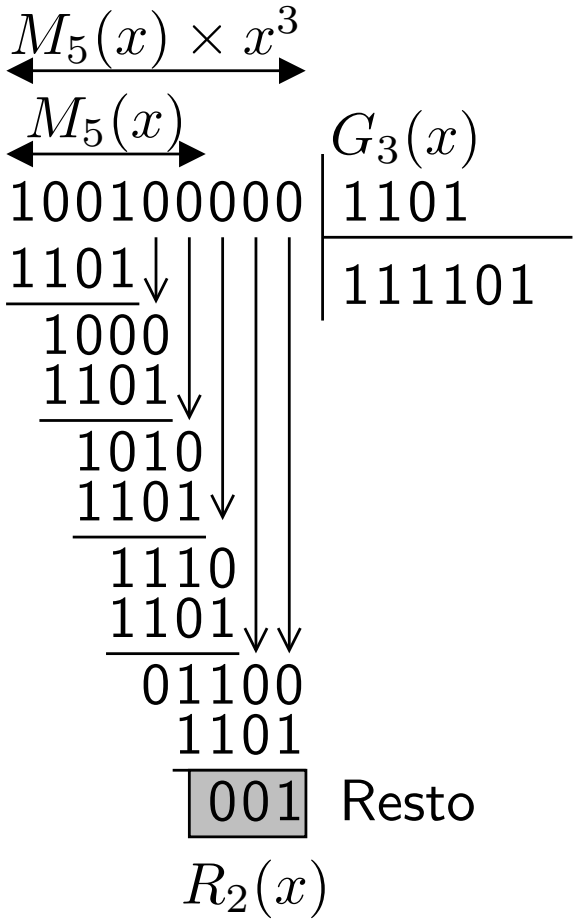
$$T_{n+k}(x) = (M_n(x) \times x^k) + R_{k-1}(x).$$

  Nótese que por la forma en que $T_{n+k}(x)$ es construído, $T_{n+k}(x)$ debe ser divisible, necesariamente, entre $G_k(x)$.

- Cuando receptor recibe $T_{n+k}(x)$, comprueba si es divisible entre $G_k(x)$. Si así es supone que no se han producido errores de transmisión.

Ejemplo

Trama a transmitir: $M_5(x) = x^5 + x^2$.
Polinomio CRC: $G_3(x) = x^3 + x^2 + 1$.
Residuo: $R_2(x) = 1$.
Trama finalmente transmitida: 100100 001
($T_8(x) = x^8 + x^5 + 1$).



Errores detectados

- Una trama con errores sería $T_{n+k}(x) + E_i(x)$, donde $E_i(x)$ es el polinomio formado por todos los bits de $T_{n+k}(x)$ que han sido invertidos por el error. Para no detectar el error tendría que ocurrir que

$$\big(T_{n+k}(x) + E_i(x)\big) \bmod G_k(x) = 0$$

o lo que es lo mismo, que

$$E_i(x) \bmod G_k(x) = 0$$

ya que por la forma en que $T_{n+k}(x)$ es construído, se cumple siempre que

$$T_{n+k}(x) \bmod G_k(x) = 0.$$

- Por tanto, los mejores polinomios generadores son aquellos que difícilmente son factor de otro posible polinomio error. Por ello, los polinomios de CRC deben: (1) tener un grado lo más alto posible y (2) no ser factorizables. Ejemplos:

Denominación	$G(x)$
CRC-8	$x^8 + x^2 + x^1 + 1$
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^1 + 1$
CRC-12	$x^{12} + x^{11} + x^3 + x^2 + 1$
CRC-16	$x^{16} + x^{15} + x^2 + 1$
CRC-CCITT	$x^{16} + x^{12} + x^5 + 1$
CRC-32	$x^{32} + x^{26} + x^{23} + x^{16} + x^{11} +$ $x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + 1$

- Nótese, por ejemplo, que el CRC-32 detectará todos aquellos errores tipo ráfaga[1] que afecte a menos de 32 bits ya que ningún polinomio error de grado menor que 32 será divisible entre él.

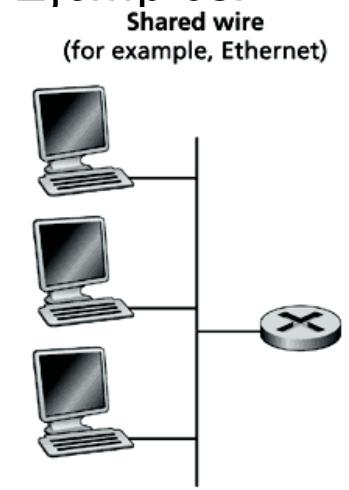
[1] Que invierte un determinado número de bits consecutivos.

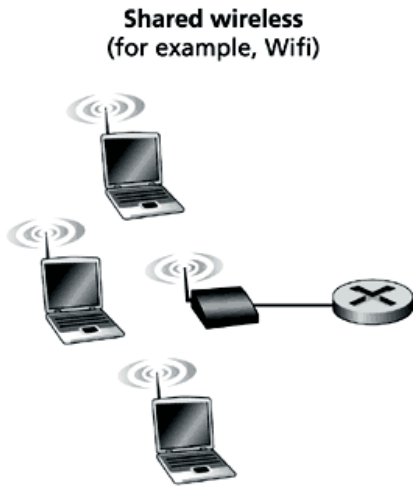
Capítulo 25

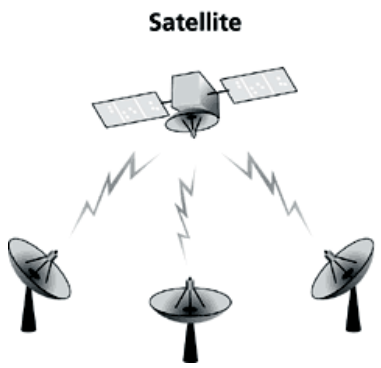
Protocolos de acceso múltiple

25.1. Protocolos de particionado del canal

- Se utilizan cuando dos o más emisores pueden acceder a un medio de transmisión compartido que no admite a más de un emisor simultáneamente [14]. Ejemplos:







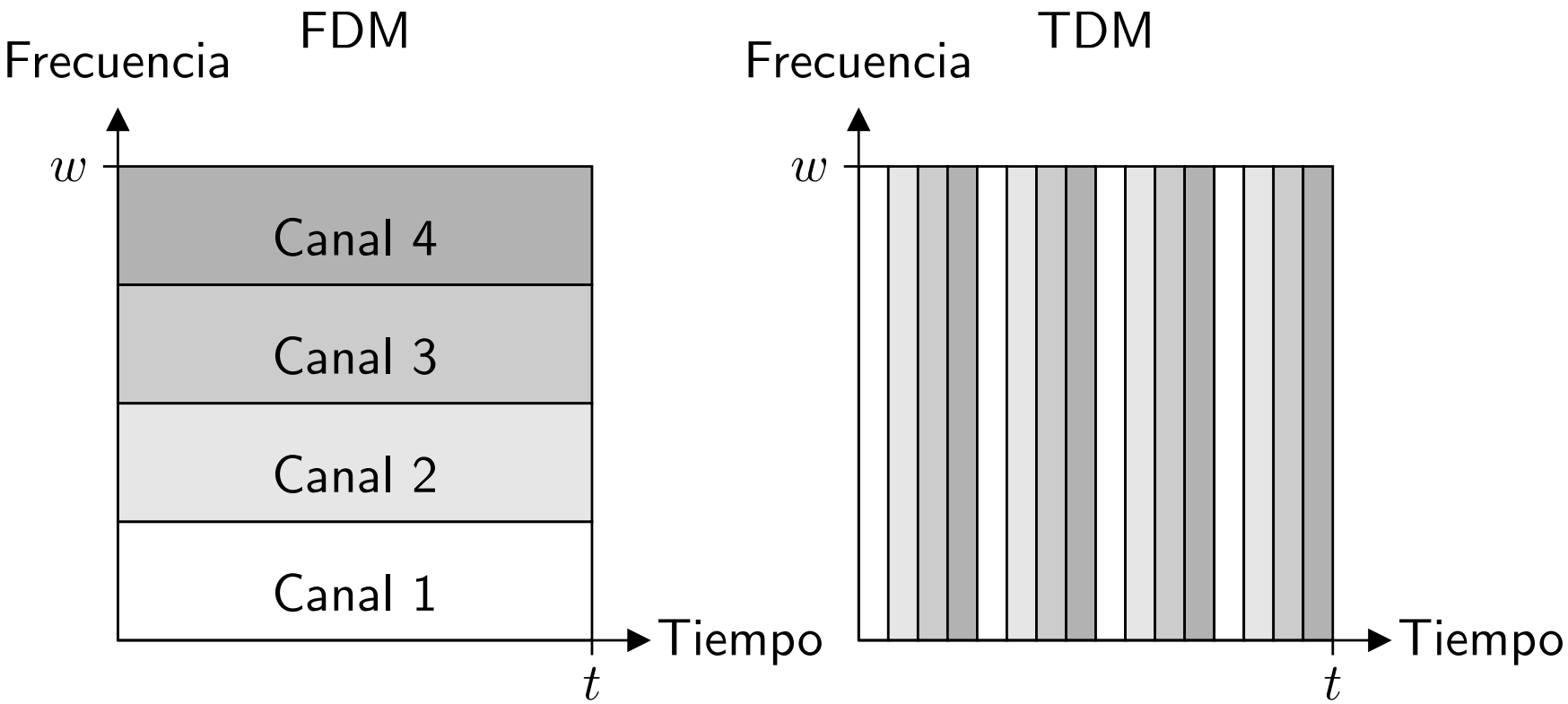
25.2. Las colisiones

- Cuando dos (o más) emisores acceden simultáneamente a un medio compartido se produce una colisión.
- La ocurrencia de la colisión implica en general que ninguno de los emisores consigue comunicarse con éxito y por tanto se desperdicia ancho de banda y/o tiempo.
- El objetivo de los protocolos de acceso múltiple consiste en evitar las colisiones.

25.3. Protocolos de particionado estático del canal

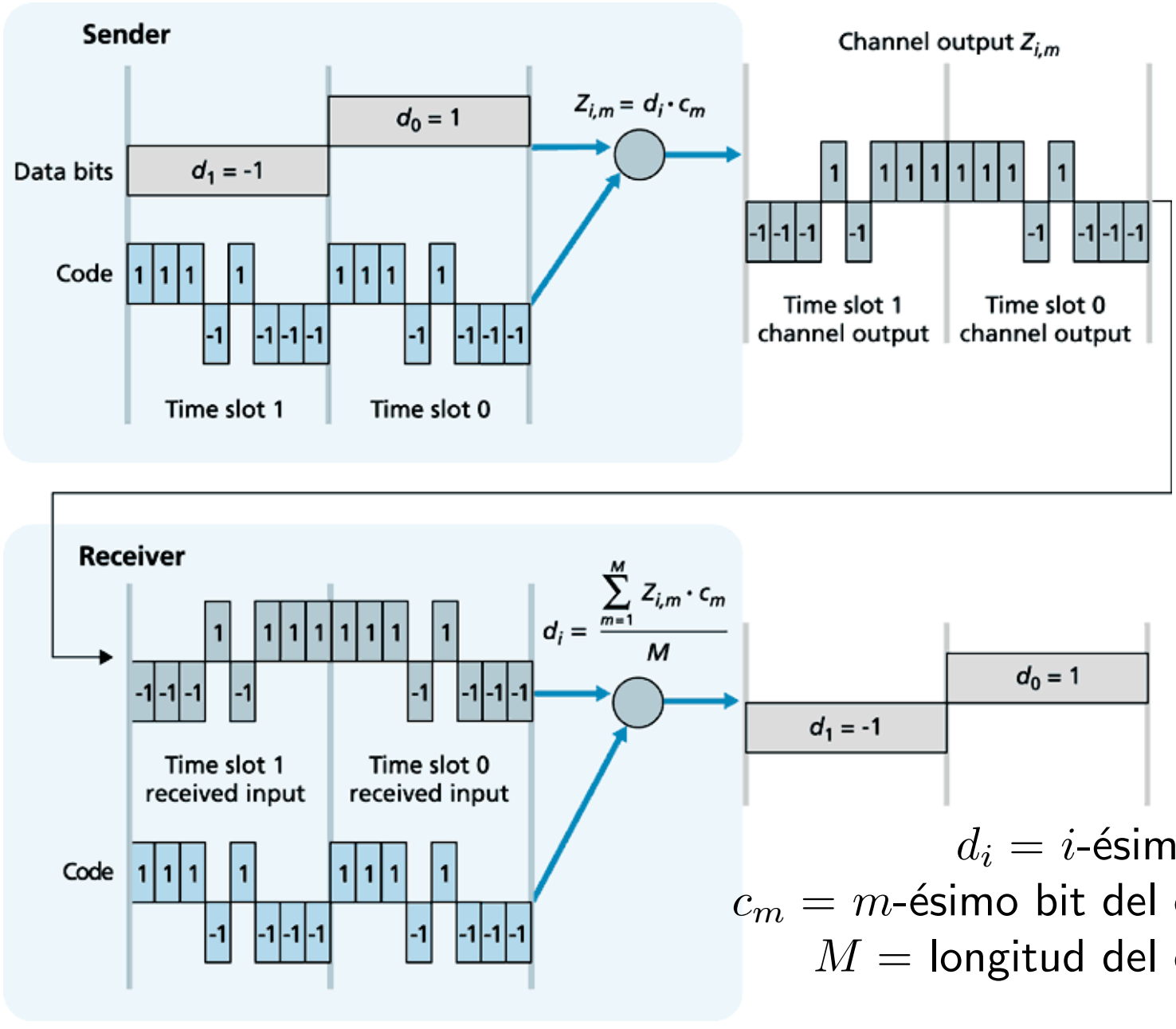
- No existen colisiones.
- Típicamente, se diseñan realizando un particionado estático del ancho de banda (FDM) o del tiempo (TDM).
- La máxima tasa de transmisión es R/N donde R es el la tasa de transmisión del enlace y N es el número de emisores.

25.3.1. FDM y TDM



25.3.2. CDMA (Code Division Multiple Access)

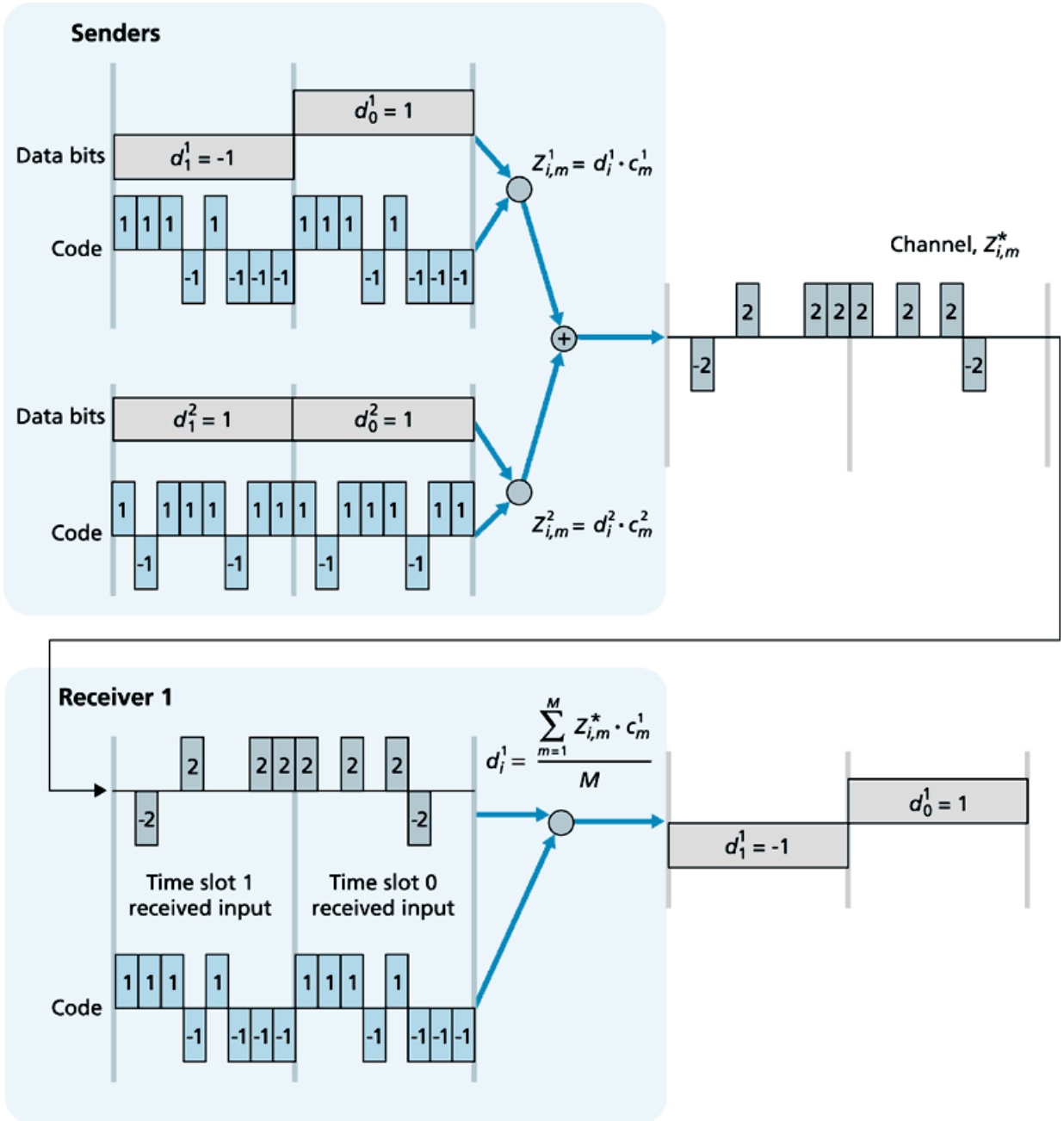
- Es semejante a FDM en que los emisores nunca cesan de transmitir, pero se transmite en banda base (sin modulación), utilizando todo el rango de frecuencias.
- Cada emisor utiliza un código específico que lo identifica y transmite cada bit de datos modulado por dicho código.
- El receptor "correlaciona" la señal recibida con el código usado por el emisor y obtiene el bit de datos.



$d_i = i$-ésimo bit de datos.

$c_m = m$-ésimo bit del código CDMA.

$M =$ longitud del código CDMA.



Las señales $Z_{i,m}$ de los emisores se suman entre sí cuando son emitidas simultáneamente, generándose $Z_{i,m}^*$.

- **Ventajas:**
  1. No hay colisiones.
  2. Alta seguridad: sólo si se conoce el código CDMA se puede recuperar la señal de datos.
- **Desventaja:**
  1. Muy alto consumo de ancho de banda (muchos baudios/bit de datos), para un emisor, dada una determinada tasa de bits. Sin embargo, como todos los emisores comparten el mismo ancho de banda, el consumo es equivalente a un TDM o un FDM.
  2. Particionamiento estático del ancho de banda.[1]

[1] Si un emisor se queda solo transmitiendo, no puede aprovecharse de esta circunstancia, al menos fácilmente, por el código CDMA depende del número potencial de emisores, no de los emisores que realmente están transmitiendo en ese momento.

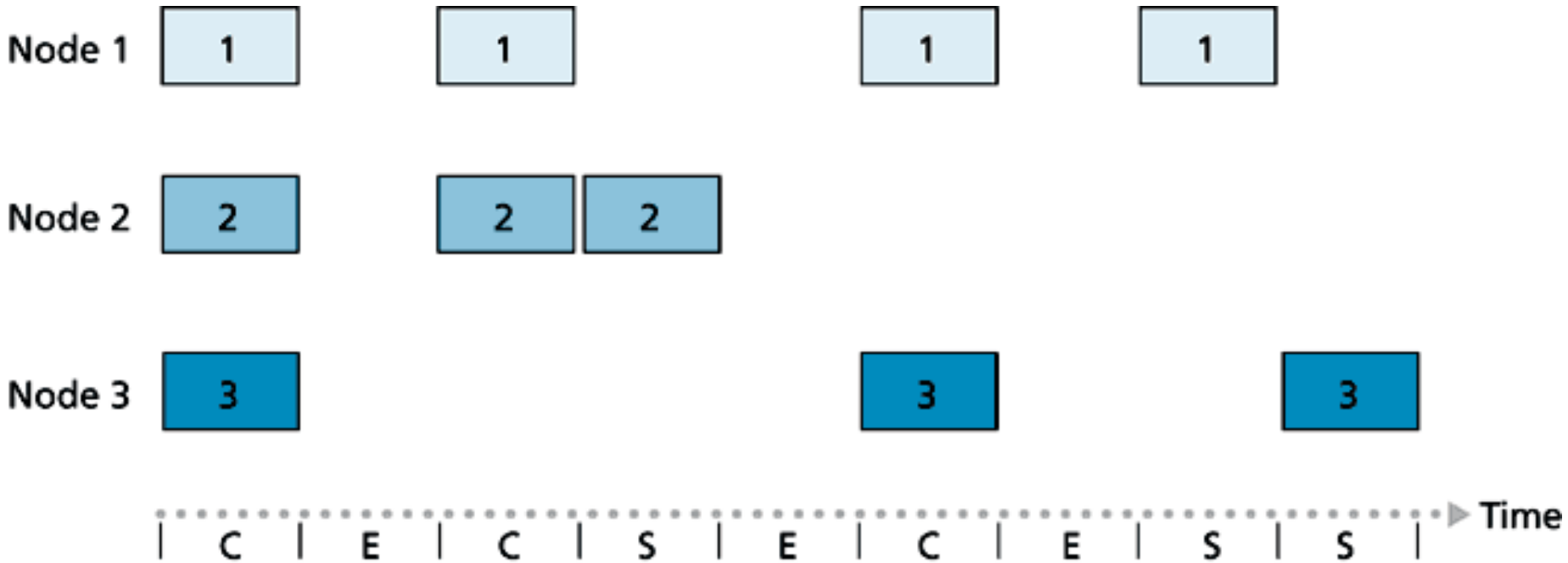
25.4. Protocolos de acceso aleatorio

- Particionado dinámico del ancho de banda.
- Existen colisiones. Cuando estas ocurren, el nodo espera un tiempo aleatorio antes de retransmitir el frame.
- La tasa de transmisión es siempre igual a R donde R es la tasa de transmisión del enlace.[2]

[2] Ojo, la tasa de transmisión efectiva, la que se obtiene útil, es típicamente inferior a R debido a las colisiones.

25.4.1. ALOHA ranurado

- Todos los frames poseen L bits.
- Se transmite en slots de tiempo de L/R segundos (1 frame/slot). Todos los nodos están sincronizados.



Key:

C = Collision slot
E = Empty slot
S = Successful slot

- Cuando se produce una colisión, **todos los nodos la detectan en ese slot de tiempo**. En ese momento todos los nodos involucrados retransmiten con una probabilidad distinta p.
- Si sólo existe un nodo (no existen colisiones), éste transmite durante todo el tiempo a R bps.[3]
- Cuando existen varios nodos transmitiendo, la eficiencia tiende a

$$N \times p \times (1-p)^{N-1}$$

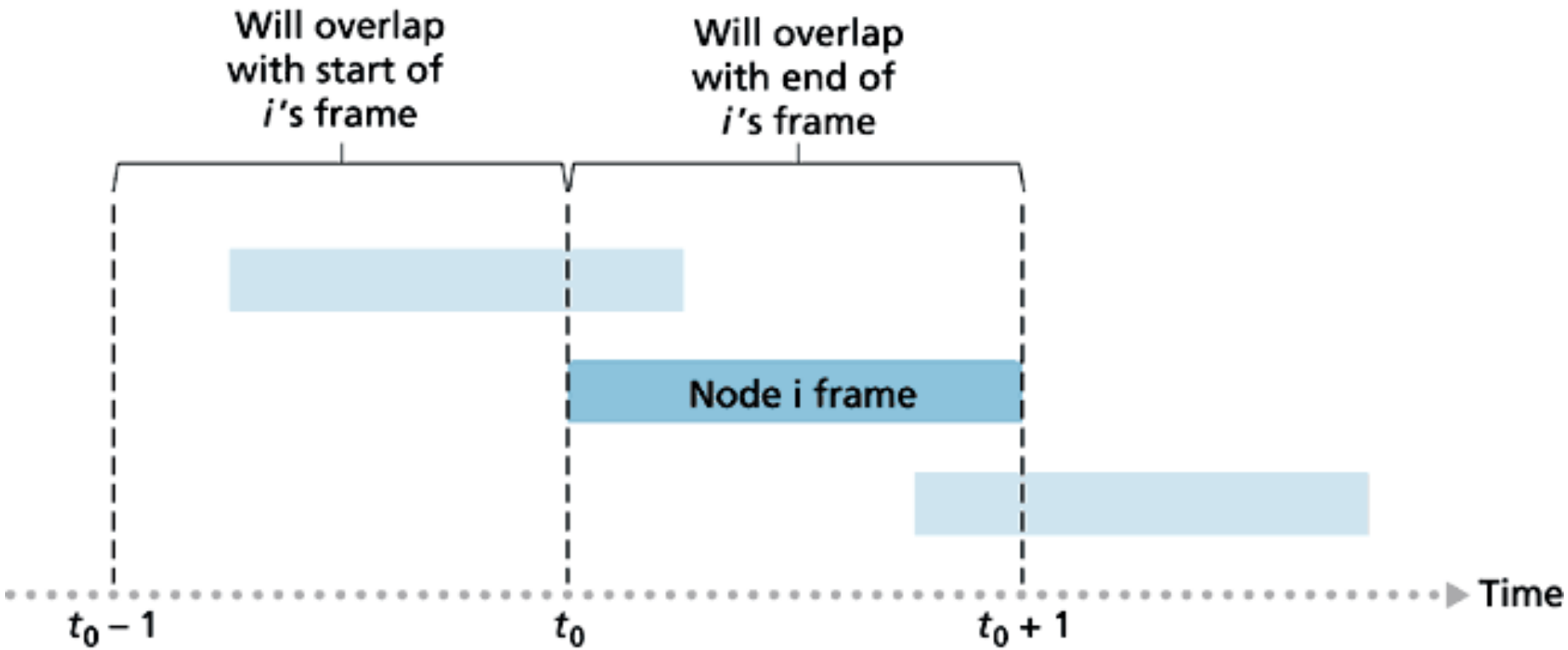
donde N es el número de nodos y p es la probabilidad de emitir un frame. De esta expresión se deduce que para un N suficientemente grande, la eficiencia es de $0{,}37$. Esto significa que sólo el $37\,\%$ de los slots de tiempo son utilizados con éxito.

- La sincronización de los nodos es crítica para evitar las colisiones parciales.

[3] Si tuviera datos que transmitir, claro.

25.4.2. ALOHA (no ranurado)

- Igual que ALOHA ranurado, pero ahora los nodos no están sincronizados.
- Siempre se transmite durante un slot de tiempo, aunque se detecte una colisión.



- La eficiencia en función del número de emisores N tiende a

$$N \times p \times (1-p)^{2(N-1)}$$

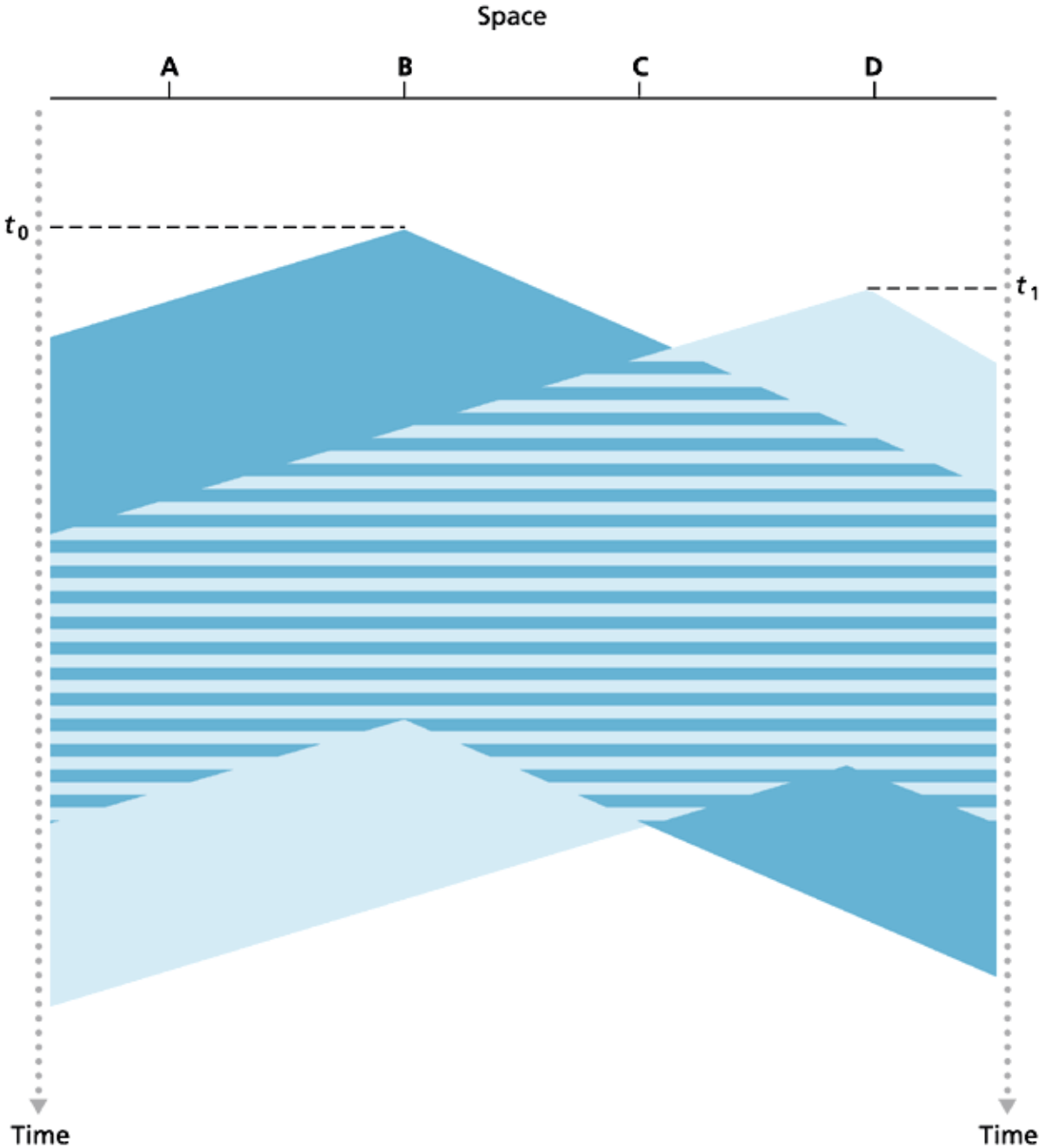
lo que significa que la eficiencia no es nunca superior a $0{,}37/2$.

25.4.3. CSMA (Carrier Sense Multiple Access)

- Incorpora la siguiente mejora (Carrier Sense):

  1. Antes de (re)transmitir, los nodos miran si el medio está ocupado. Si lo está, esperan un tiempo aleatorio.

- A pesar de esta mejora, pueden existir colisiones. Supongamos, por ejemplo, que existen 4 nodos A, B, C y D situados a lo largo de un cable como se muestra en la siguiente figura.



En t_0 emite B
y en t_1 emite D.
Si se cumple que
en $t_1 - t_0$ la señal
portadora no ha llegado
desde B hasta D,
entonces se producirá
una colisión.

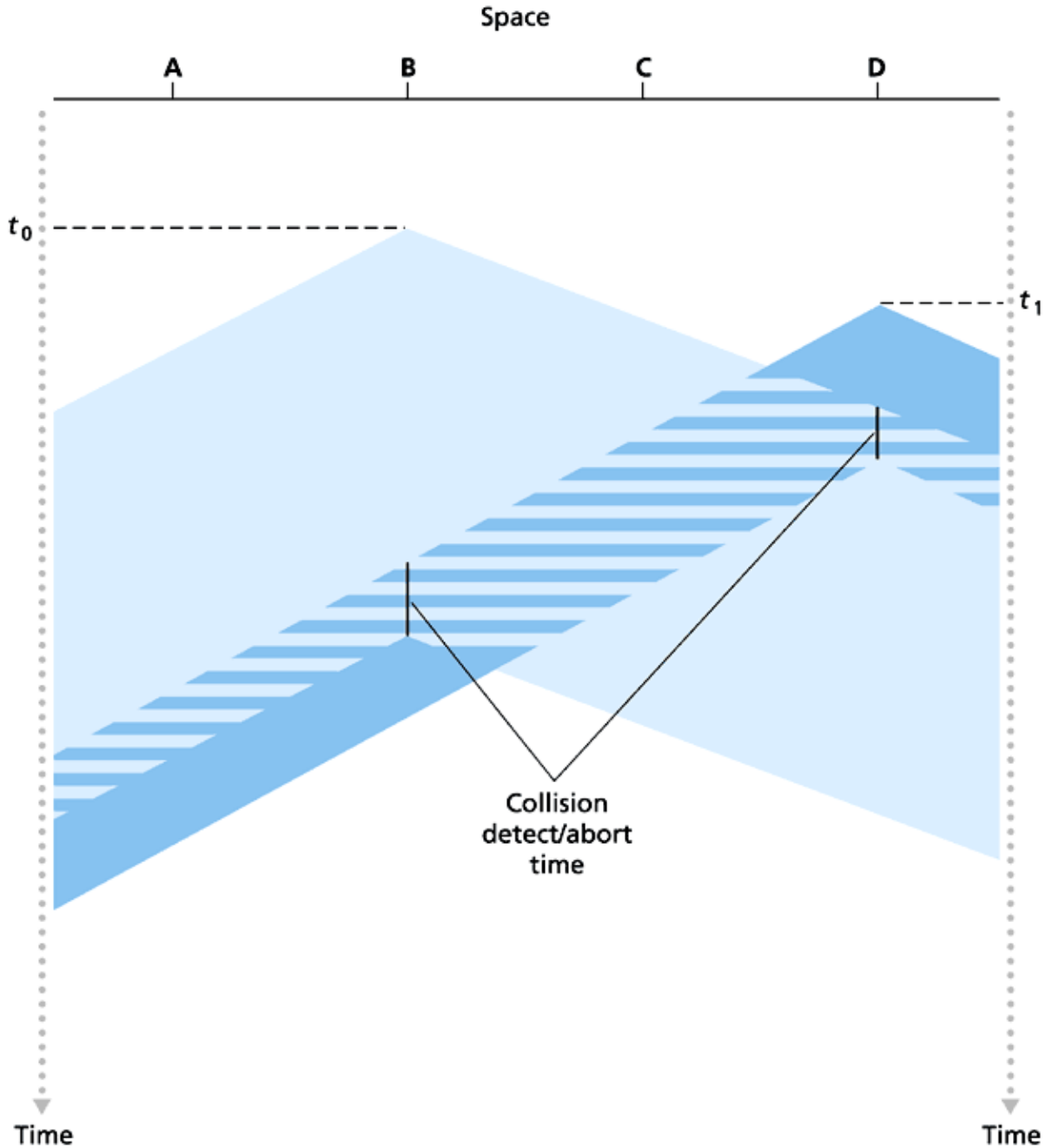
25.4.4. CSMA/CD (Collision Detect)

- Incorpora la siguiente mejora (Collision Detect):

  1. Mientras transmiten, si detectan una colisión entonces dejan inmediatamente de transmitir.

- Usado en Ethernet.

- Las colisiones pueden seguir apareciendo, pero ahora el tiempo de colisión es menor. En el siguiente ejemplo (semejante al anterior) se ha supuesto que el tiempo que se necesita para detectar la colisión no es cero (lo que en la práctica siempre ocurre).


Space
A
B
C
D
t_0
t_1
Collision
detect/abort
time
Time
Time

25.5. Protocolos basados en turnos

- Se basan en establecer alguna política de acceso al medio basada en turnos. Se han llevado a la práctica dos alternativas diferentes:

  1. **Protocolos de sondeo**. Un "nodo maestro" se encarga de ir chequeando (secuencialmente) qué nodos tienen datos que transmitir. Cuando detecta uno le da permiso durante un tiempo limitado.
  2. **Protocolos de paso de token**. Los nodos se intercambian (secuencialmente) un frame especial llamado *token* que permite al nodo que lo posee transmitir durante un tiempo limitado.

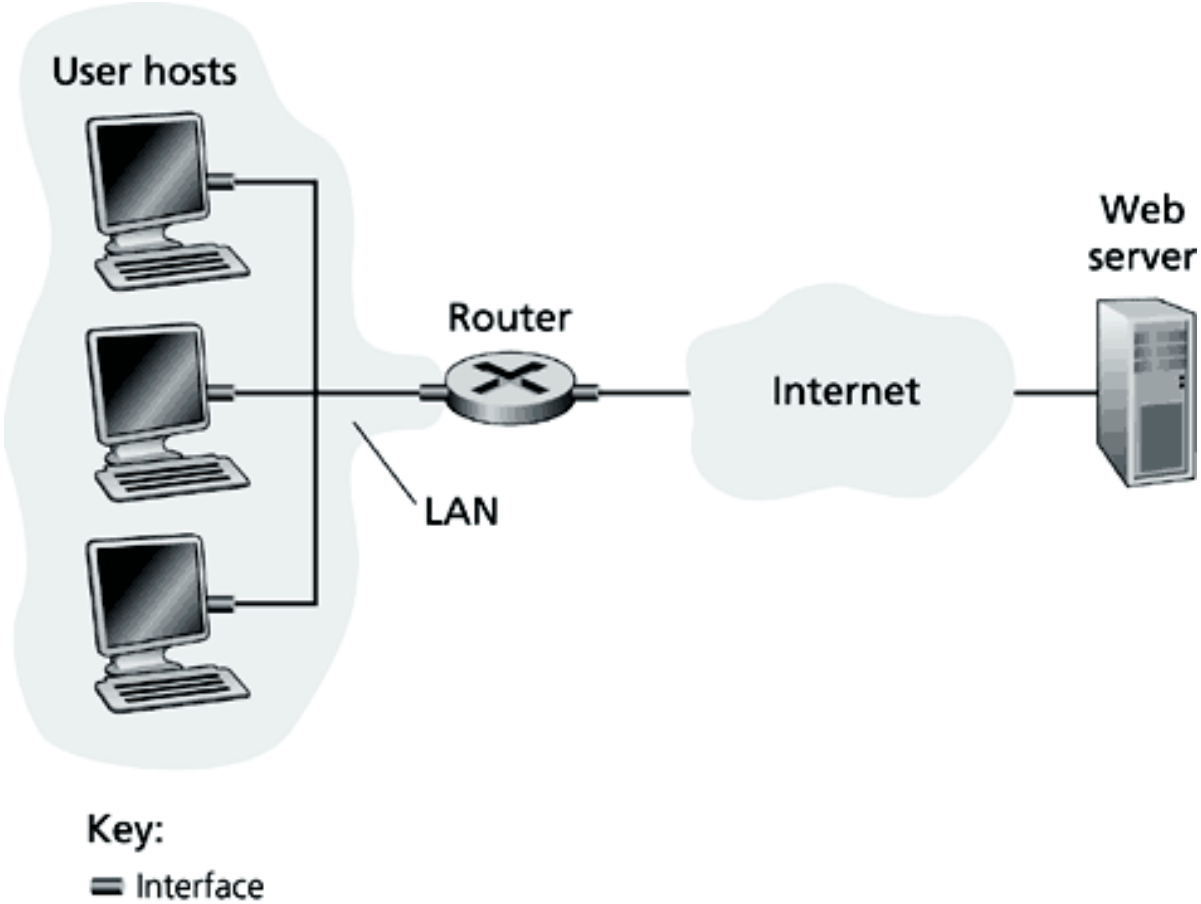
- Los protocolos basados con turnos poseen una eficiencia mayor que los protocolos de acceso aleatorio cuando el número de nodos potencialmente emisores es suficientemente alto (ya que no existen colisiones). Sin embargo, cuando hay pocos emisores, la eficiencia es menor porque el sondeo o el paso del token no es instantáneo.

Capítulo 26

Redes locales y el ARP

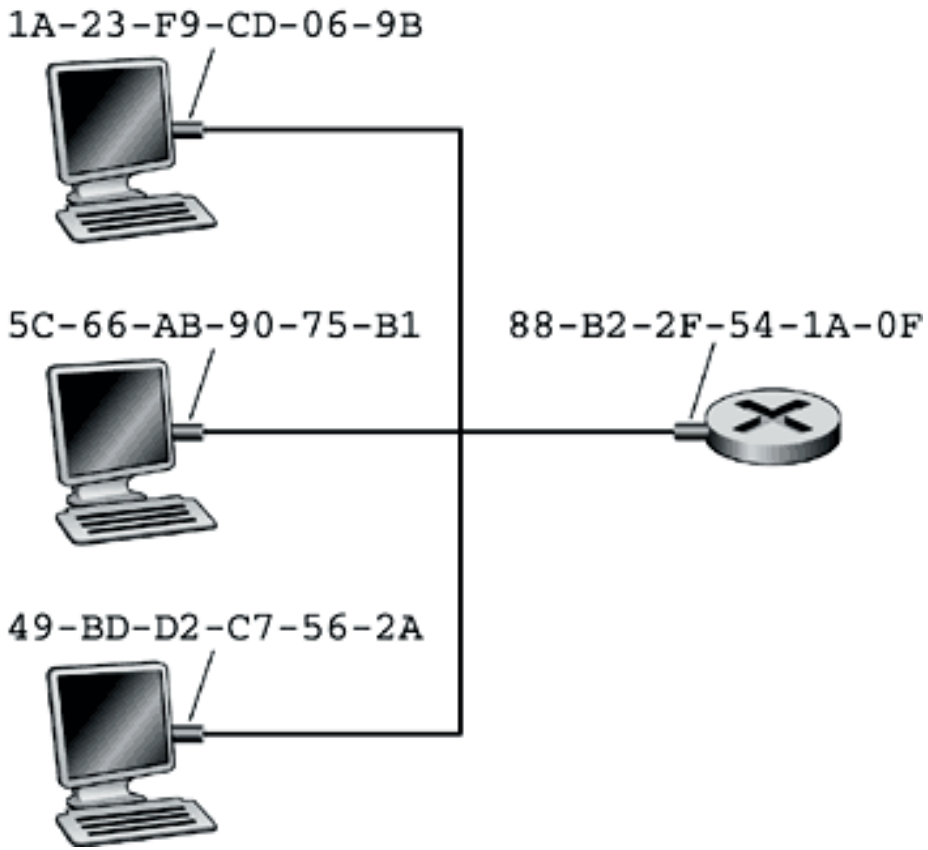
26.1. LAN: definición

- Una LAN (Local Area Network) es una red de computadoras concentradas en un área geográfica [14].
- Una LAN está típicamente conectada a Internet a través de un router.



26.2. Direcciones físicas

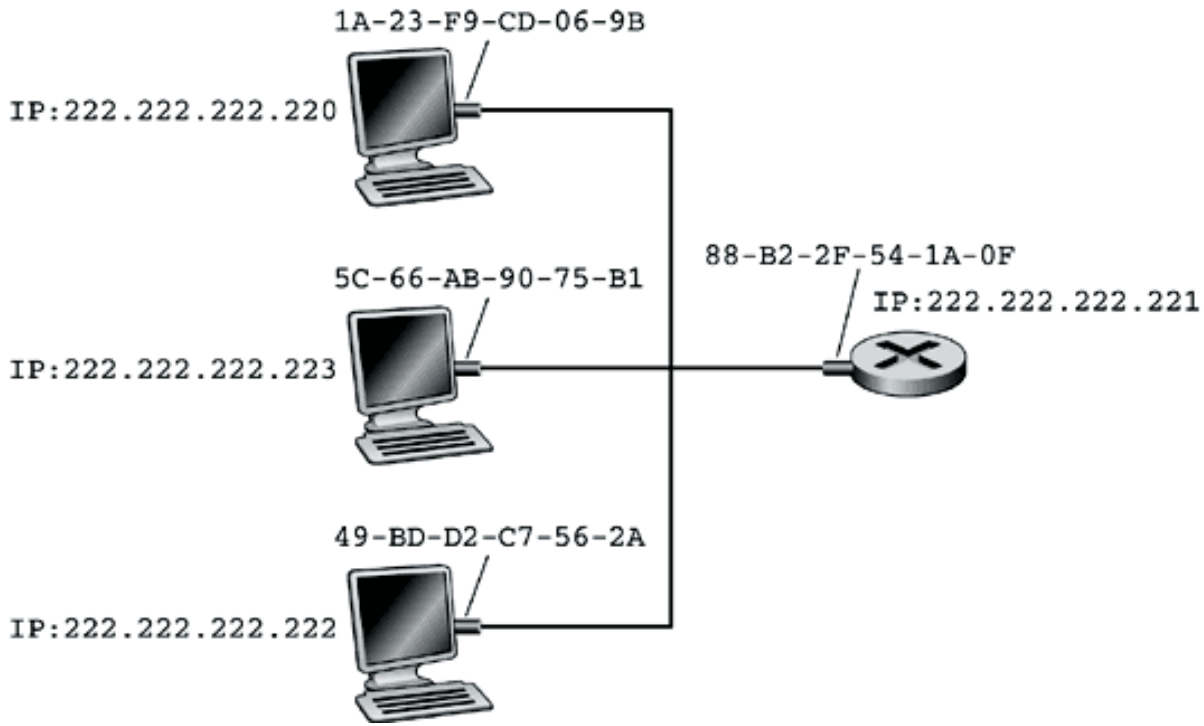
- Cada adaptador de red posee una dirección física (también llamada dirección MAC – Medium Access Control –).
- En la mayoría de las tecnologías de red, las direcciones físicas poseen 6 bytes.



- En todos los frames transmitidos figura la dirección física del adaptador de red destino.
- La dirección física FF-FF-FF-FF-FF-FF es la dirección de broadcast de la sub-red y provoca que el frame alcance a todos los adaptadores conectados.

26.3. El ARP (Address Resolution Protocol)

- RFC 826.
- Es utilizado por todos los nodos que poseen la capa de enlace de datos.
- Traduce la dirs IP a dirs físicas.



- Cada nodo incorpora una tabla ARP:

Dir IP	Dir Física

- Las tablas ARP son generalmente dinámicas. Los nodos están siempre pendientes de los frames que les llegan y utilizan las parejas de dirs IP/dirs físicas (que en ellos figuran) para mantenerlas lo más actualizadas posible. Además, sus entradas tienen un tiempo máximo de vida.

- Cuando la tabla ARP no contiene una resolución se ejecuta el siguiente protocolo (ARP):

  1. Enviar un frame ARP a la dir de broadcast de la sub-red. Dicho frame contiene la dir física del nodo emisor y la dir IP del nodo receptor.
  2. Cada nodo recibe el frame ARP y comprueba si su dir IP concuerda con la que figura en el frame. Si así es, contesta (usando otro frame ARP) a la dir física del nodo que quiere enviar el frame de datos. De paso actualiza su tabla ARP.
  3. El nodo que quiere enviar el frame de datos recibe el segundo frame ARP con la resolución (en él figura la dir física del nodo que lo genera), actualiza su tabla ARP y transmite los datos.

Capítulo 27

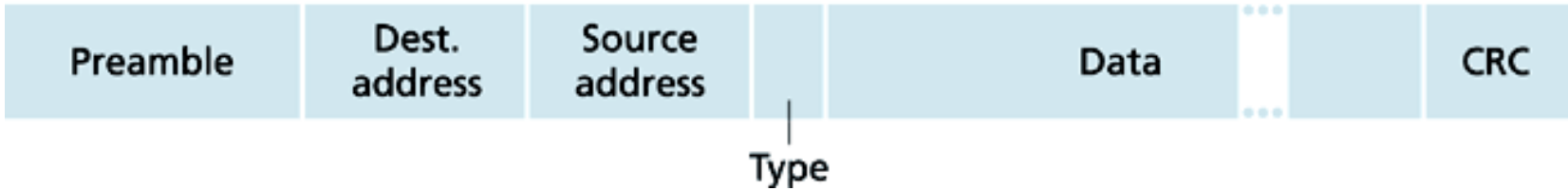
Ethernet

27.1. Historia

- La tecnología Ethernet fue inventada por Metcalfe y Boggs a mediados de los 70 [14].
- Fue la primera en su género y actualmente es la tecnología predominante en redes de área local (cableada).

27.2. Estructura del frame

- Todas las tecnologías Ethernet utilizan la misma estructura de frame[1]:



1. **Preámbulo** (8 bytes): consta de XXXX XXXY donde X = 1010 1010 e Y = 1010 1011. Se utiliza para sincronizar los relojes del emisor y del receptor que miden la duración de los bits del frame.
2. **Dir. física destino** (6 bytes): dir del adaptador destino.
3. **Dir. física fuente** (6 bytes): dir del adaptador origen.
4. **Tipo** (2 bytes): identifica el protocolo usado en la capa de red (IP, Novell IPX, AppleTalk, etc.).

[1] Esto ha permitido que el software de la capa de red haya sido reutilizado durante más de 20 años.

5. **Datos** (entre 46 y 1.500 bytes): paquete de datos transportado. La capa de red se encarga de la segmentación y del relleno con ceros (si estos fueran necesarios).

6. **CRC** (4 bytes): CRC-32. Sirve para detectar errores de transmisión.

27.3. Tamaño máximo y mínimo de frame

- Ethernet es incapaz de encapsular un paquete proveniente de la capa de rez que ocupe más de 1.500 bytes. Este valor es suficientemente bajo como para que los tiempos de espera de los adaptadores que están esperando a transmitir no sea excesivo y suficientemente alto como para que el overhead de las cabeceras sea reducido.

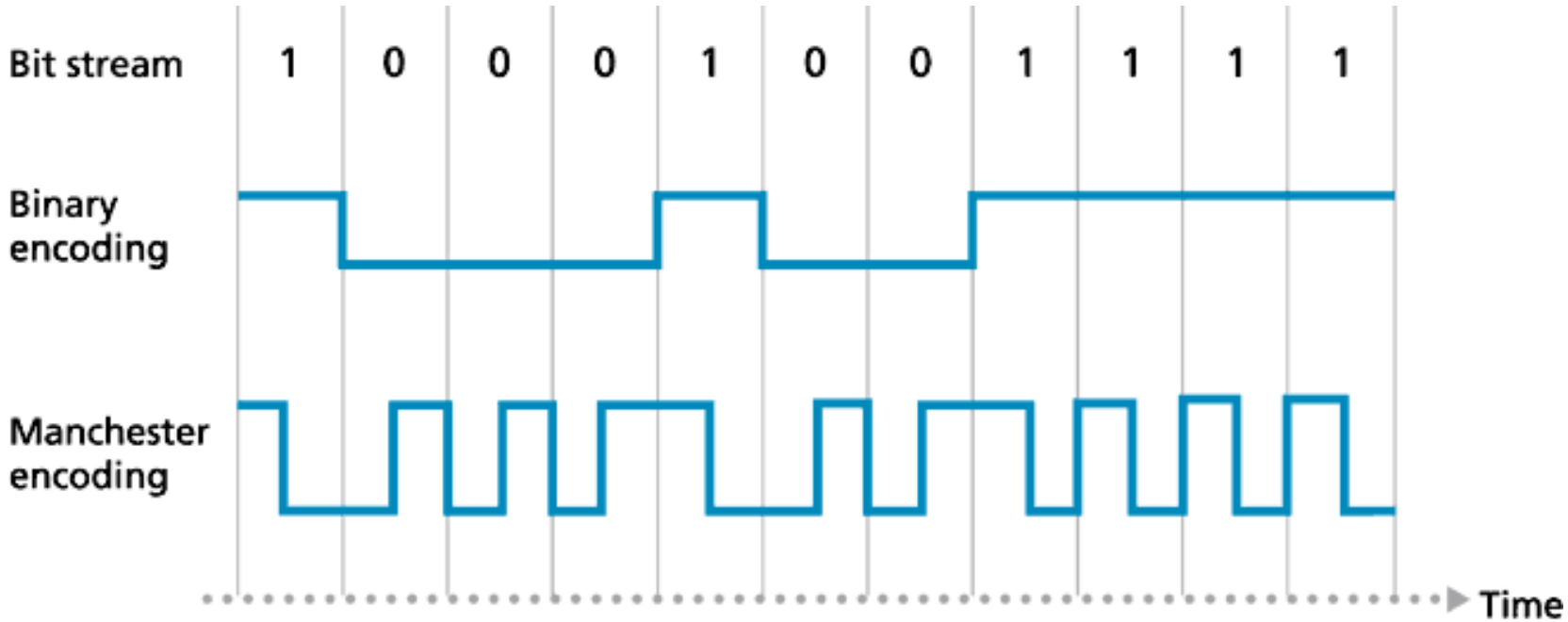
- Ethernet tampoco puede transportar payloads de menos de 46 bytes por la siguiente razón. Si sumamos los 18 bits de cabecera que transportan información (todos los campos excepto los bits de sicronismo) resultan 64 bytes (512 bits). La Ethernet original podía extenderse hasta 200 metros y presentaba un RTT de 50 micro-segundos, aproximadamente. En la Ethernet original la tasa de transmisión era 10 Mbps y a esta tasa de bits, 512 bits tardan en inyectarse en el medio 51,2 micro-segundos. Por tanto, cualquier estación estaría transmitiendo aún si ella hubiera provocado una colisión. Así, sabría que tiene que volver a retransmitir el frame.

27.4. Servicio

- Ethernet posee las siguientes características:
  1. **Sin conexión**: los frames Ethernet se transmiten sin previo aviso.
  2. **No fiable**: si un frame llega con errores, Ethernet únicamente los desecha.
- Es responsabilidad de las capas superiores transformar este servicio en fiable y orientado a conexión (cuando sea preciso).

27.5. Codificación Manchester

- Se utiliza transmisión en banda base, utilizando codificación Manchester:



- Una tasa de baudios igual al doble de la tasa de bits permite mantener los relojes del emisor y del receptor siempre sincronizados. Sin embargo, esto genera un mayor consumo de ancho de banda (el doble que si no se usa dicha sincronización).

27.6. El protocolo CSMA/CD en Ethernet

- El adaptador de red intenta transmitir cada frame lo antes posible, pero primero comprueba (durante 96 tiempos de bit) a que no exista señal "portadora" en el medio.
- Si durante una transmisión no se detecta una colisión, se supone que el frame se ha transmitido con éxito.[2] Sin embargo, si se detecta una colisión la transmisión se aborta inmediatamente y se transmite una señal de *jam* (atasco) de 48 bits iguales a 0.[3]

[2] Por esto los frames tienen que tener un tamaño mínimo de 512 bits, sin contar el preámbulo (64 bits).

[3] Esta señal avisa al resto de emisores que se ha producido una colisión.

- Todos los adaptadores que han producido una colisión ejecutan el **algoritmo de retroceso exponencial binario**. Este consiste en:

  1. Sea $n \leftarrow 1$ el número de colisiones experimentadas en la transmisión del frame en cuestión.
  2. Mientras persista la situación de colisión:
     a) Generar un número entero aleatorio uniforme $K \in \{0, 1, \cdots, 2^m - 1\}$, donde $m = \text{mín}(n, 10)$.
     b) Esperar $K \times 512$ tiempos de bit antes de retransmitir.
     c) $n \leftarrow n + 1$.
     d) Si $n \geq 16$ abortar transmisión.

27.6.1. Eficiencia

- Cuando sólo existe un nodo emisor, la eficiencia máxima (teórica) en Ethernet es 1 ($100\,\%$).[4]
- Sin embargo, en la práctica se ha comprobado que ésta tiende a:

$$\frac{1}{1 + 5t_{prop}/t_{trans}}$$

  donde t_{prop} es igual al máximo tiempo de propagación de una señal entre los adaptadores más distantes y t_{trans} es el tiempo de transmisión (inyección en o extracción del enlace) del frame más largo.
- Como ejemplo, para un enlace Ethernet de 100BaseT de 500 m, una velocidad de propagación de 250000 Km/s, y sabiendo que el tamaño máximo de frame en Ethernet es de 1526 B, resulta una eficiencia del $90\,\%$, aproximadamente.

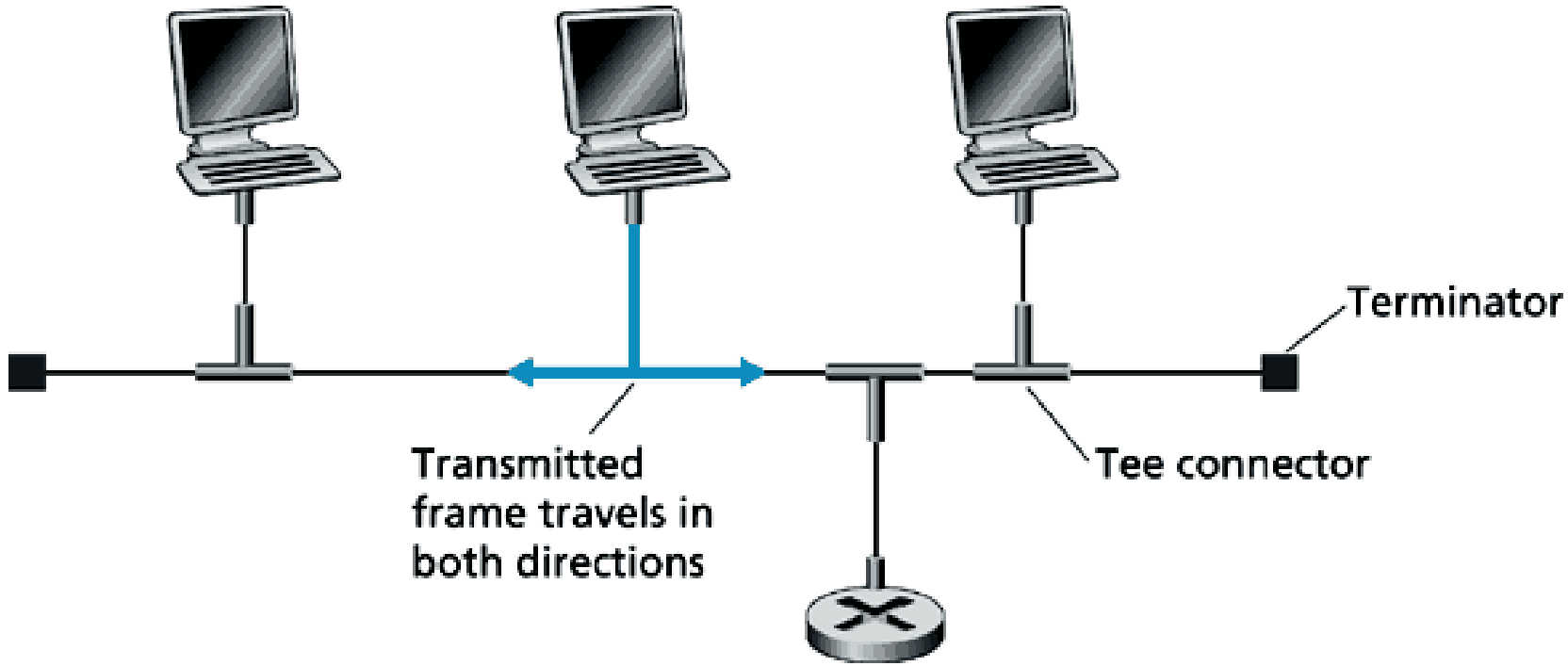
[4] La eficiencia real depende de una casuística concreta y siempre va a ser menor.

27.7. Tecnologías Ethernet

- Estandarizadas por el grupo de trabajo IEEE 802.3.

27.7.1. 10Base2 Ethernet

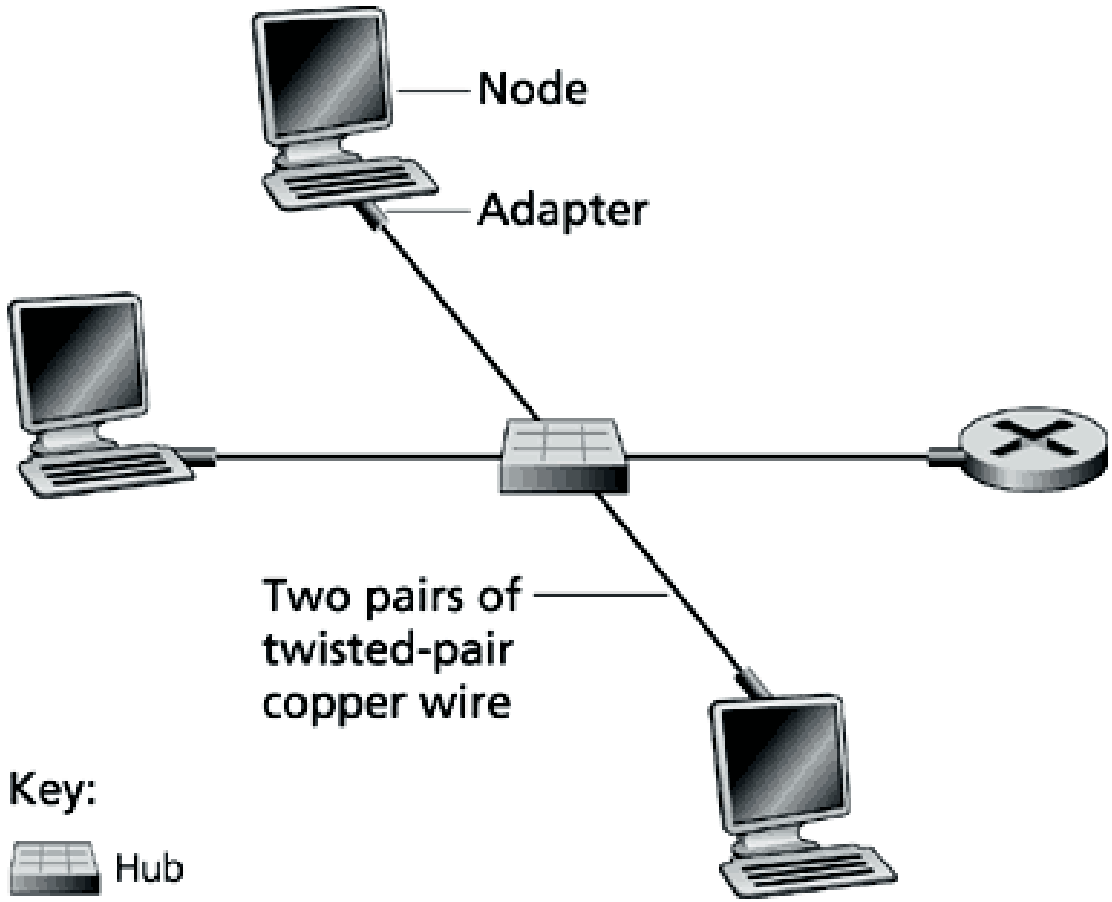
- Creada a principios de los 90.
- Consigue 10 Mbps usando transmisión en banda base y permite enlaces de hasta 200 metros de longitud (de ahí el "2" en su nombre).
- Los adaptadores se conectan entre sí usando cable coaxial delgado y conectores en T.



- Se trata de una tecnología donde sólo existe un dominio de colisión. Los frames se propagan en ambas direcciones, alcanzan a todos los adaptadores, van perdiendo energía en su viaje y finalmente mueren (se atenúan totalmente) en los terminadores (resistencias de $50\ \Omega$).

27.7.2. 10BaseT Ethernet y 100BaseT Ethernet

- Es la más usada actualmente.
- Funcionan (respectivamente) a 10 Mbps y a 100 Mbps, transmiten en banda base y utilizan (típicamente) par trenzado (T).
- La topología típica de interconexión es en estrella. Todos los adaptadores se conectan a través de un concentrador (un hub o un switch) mediante enlaces punto a punto de dos pares (full-duplex) trenzados y terminados en conectores RJ-45.



- Dependiendo de si el concentrador es un hub o un switch, existen uno o muchos dominios de colisión.
- Usando par trenzado de cobre no suele ser posible separar más de 100 metros el concentrador de ninguno de los nodos. Para mayores

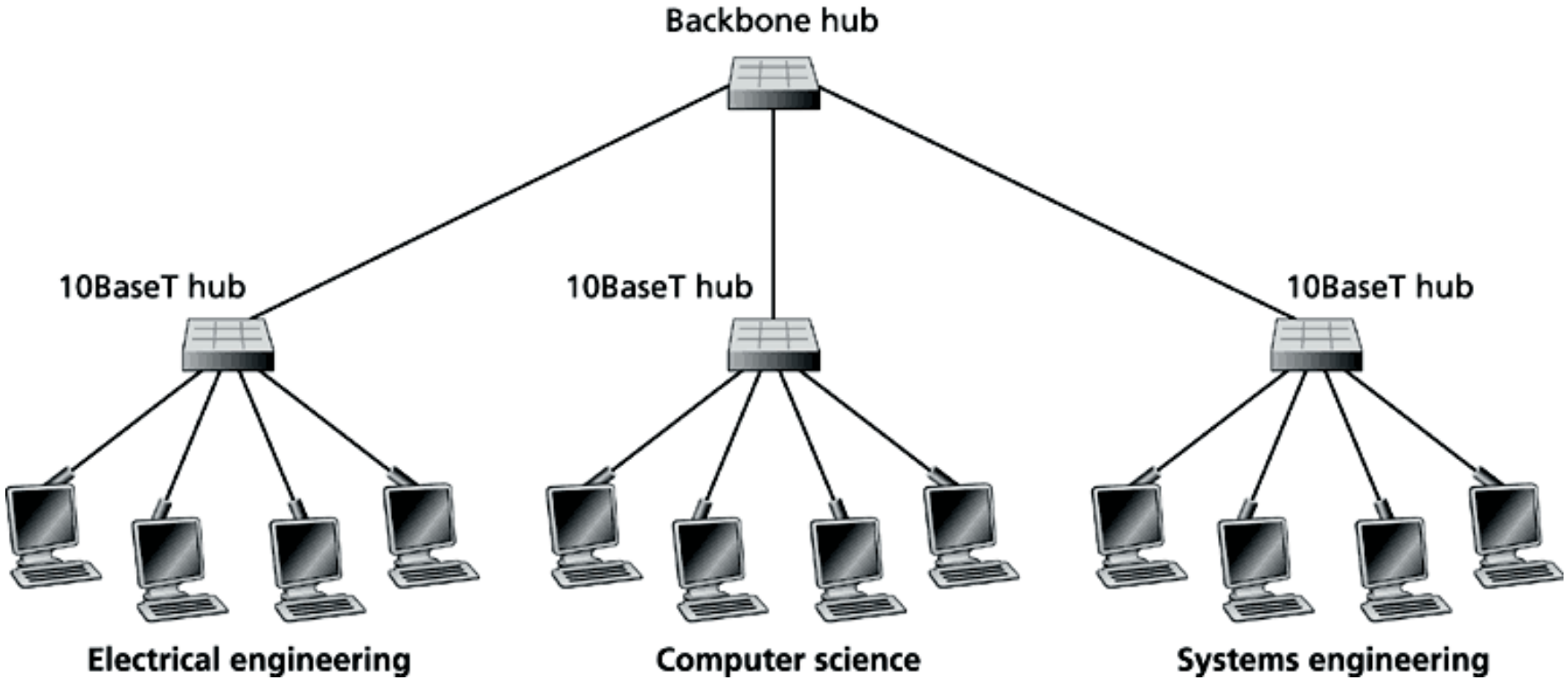
distancias se puede utilizar fibra óptica que es mucho menos sensible al ruido.

27.7.3. Gigabit Ethernet y 10 Gigabit Ethernet

- Son mejoras de (y compatibles con) las normas 10BaseT y 100BaseT que permiten alcanzar 1 Gbps y 10 Gbps, respectivamente.

27.8. Hubs

- Los hubs son básicamente repetidores (dispositivos de regeneración de señales digitales que funcionan a nivel de la capa física).
- Suelen ser multipuerto: la señal que llega a través de alguna de sus entradas es propagada (tras ser despojada del ruido) a todas las salidas [14].



- No entienden los protocolos de la capa de enlace de datos (ni por supuesto los superiores).
- No crean dominios de colisión, sólo los extienden. En este sentido todos los nodos que se conectan mediante un hub acceden al medio usando CSMA/CD.

27.9. Switches

- Los *conmutadores Ethernet* son dispositivos de conmutación de frames que funcionan a nivel de la capa de enlace de datos.
- A diferencia de los *hubs*, pueden interconectar diferentes tecnologías Ethernet.
- También a diferencia de los *hubs*, separan los dominios de colisión. Por ejemplo, en la siguiente red existen 3 dominios de colisión diferentes, uno para cada departamento:

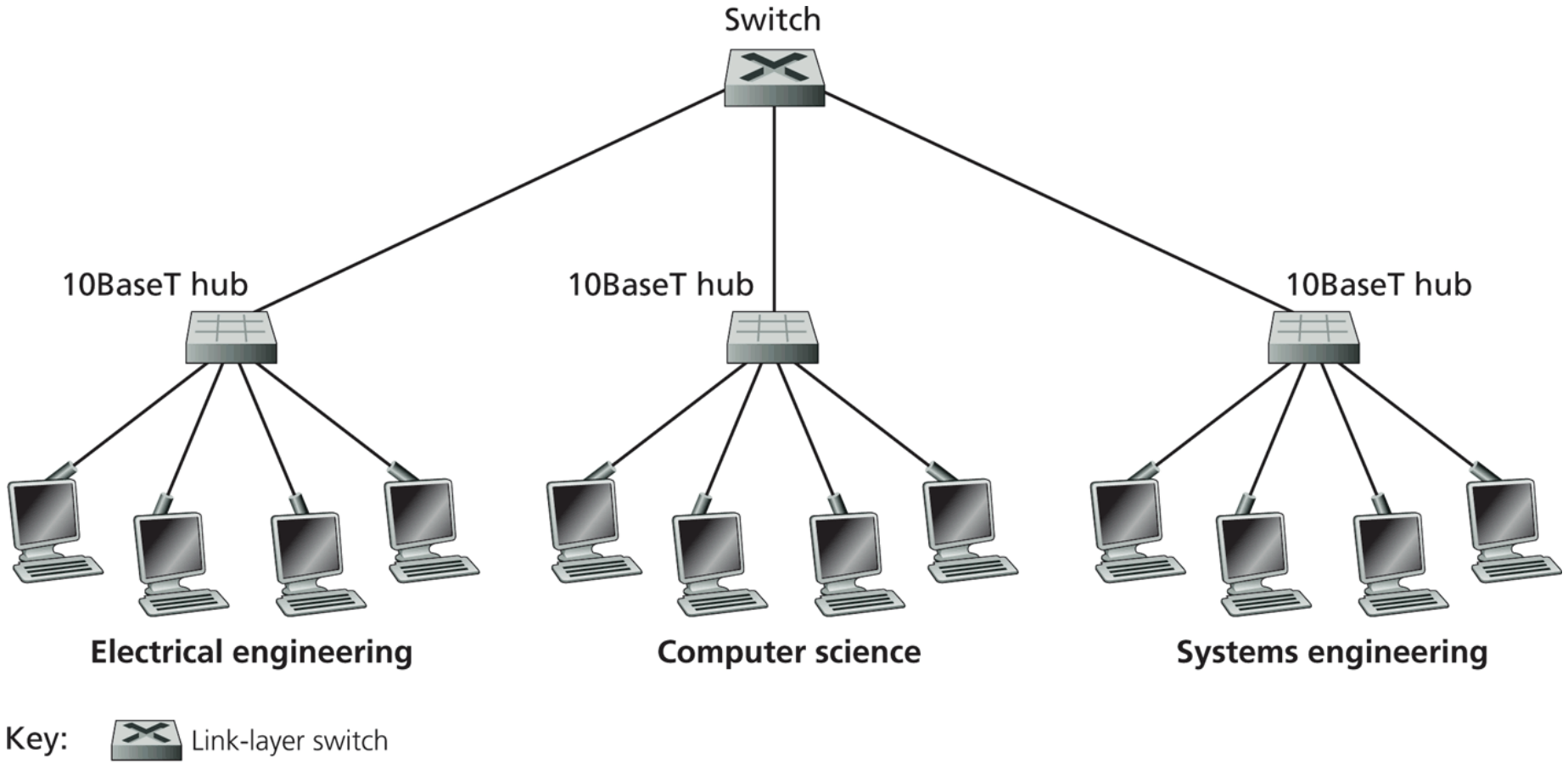


Figure 5.28 ♦ Three departmental LANs interconnected with a switch.

- Otro ejemplo:

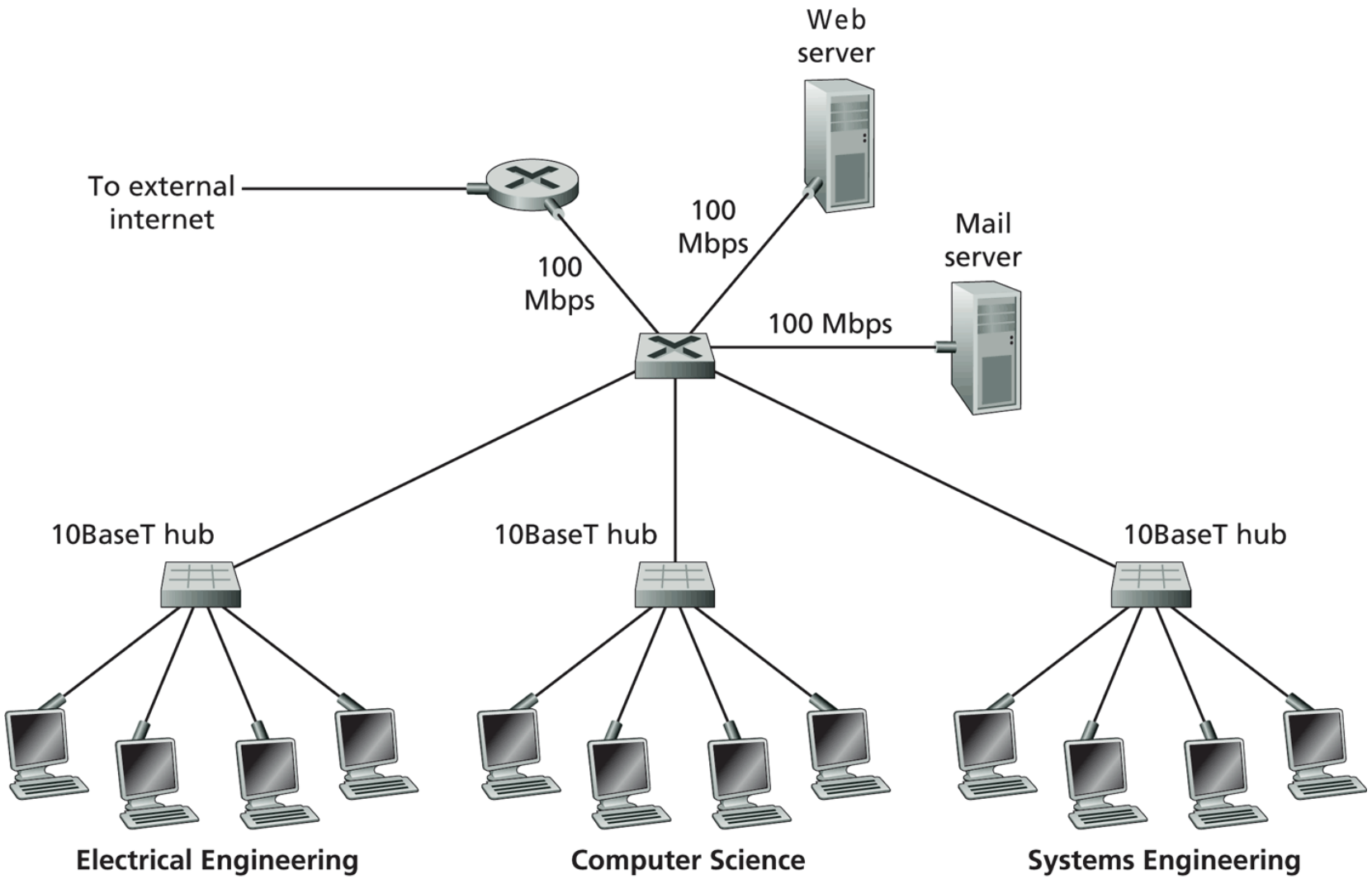


Figure 5.29 ♦ An institutional network using a combination of hubs, Ethernet switches, and a router

- Como se desprende de los anteriores ejemplos, los switches deben ejecutar el protocolo CSMA/CD sólo cuando en enlace involucrado no sea full-duplex.

27.9.1. Encaminamiento

- Cada switch posee una *switch table* con la estructura

Address	Interface	Time
62-FE-F7-11-89-A3	1	9:32
7C-BA-B2-B4-91-10	3	9:36
....	....	....

Figure 5.30 ♦ Portion of a switch table for the LAN in Figure 5.28

donde la primera columna indica la dirección física de los diferentes interfaces de red conectados al switch (directamente o a través de un hub), la segunda es la boca al que se conecta cada interface y la tercera, el último instante en que se ha escuchado a ese interface.

- La tabla de conmutación es dinámica. El switch la mantiene de forma automática anotando la dir física origen de cada frame que le llega. Cuando una entrada se hace demasiado vieja, se elimina.
- Así, el algoritmo de encaminamiento consiste en:

1. Buscar en la *switch table* el interface de salida asociado a la dir física destino del frame.
2. Si existe una entrada para la dir física destino, entonces:
   a) Si el interface de salida es distinto del de entrada (piénsese en que en un interface pueden conectarse más de un adaptador de red usando un hub), entonces:
      1) Encaminar el frame por el interface de salida.
3. Si no existe ninguna coincidencia:
   a) Realizar un flooding. Los switches ejecutan el algoritmo del árbol de expansión para evitar un broadcast-storm.

27.9.2. Store-and-forward versus cut-through

- Cuando las tasas de transmisión de la boca de entrada de un frame coincide con la tasa de transmisión de la boca de salida, el switch puede usar una técnica de transferencia conocida como cut-through y que consiste en que los frames se transmiten mientras todavía se están recibiendo.
- Esto permite reducir sensiblemente la latencia, especialmente cuando los frames necesitan atravesar muchos switches.

Capítulo 28

Wi-Fi

28.1. Distancias

- La distancia a recorrer depende del tipo de antenas usadas, de las potencias de emisión y de los objetos que puedan existir entre las antenas.
- La señales de radio Wi-Fi se reflejan (sobre todo en superficies metálicas) y se refractan.
- Sin obstáculos y usando antenas unidireccionales (parabólicas o planas) las distancias pueden ser de miles de metros.
- Dentro de las casas, y usando antenas omnidireccionales (bobinas o dipolos) las distancias sueles ser unos pocos metros.

28.2. Capacidades

Norma	Banda de Frecuencias	Mutiplexación	Tasa de Transmisión
802.11a	5,1 GHz - 5,8 GHz	OFDM	Hasta 54 Mbps
802.11b	2,4 GHz - 2,485 GHz	DSSS	Hasta 11 Mbps
802.11g	2,4 GHz - 2,485 GHz	OFDM	Hasta 54 Mbps

OFDM = Orthogonal Frecuency-Division Multiplexing.
DSSS = Direct Sequence Spread Spectrum.

- Wireless LAN IEEE 802.11. Wi-Fi es simplemente una norma de calidad (como THX o HiFi para los equipos de audio).
- Uso creciente porque permite una movilidad total en distancias medias (una casa, por ejemplo).
- Todas utilizan el mismo protocolo de acceso al medio, el CSMA/CA, y la misma estructura de frame.

28.3. Modes

28.3.1. Infrastructure

- Es el modo más frecuente.
- El AP (Access Point) permite que los nodos móviles se comuniquen entre sí y con la red "wired".
- Los nodos móviles junto con el AP correspondiente forman un BSS (Basic Service Set).
- Todas las comunicaciones pasan a través del AP.

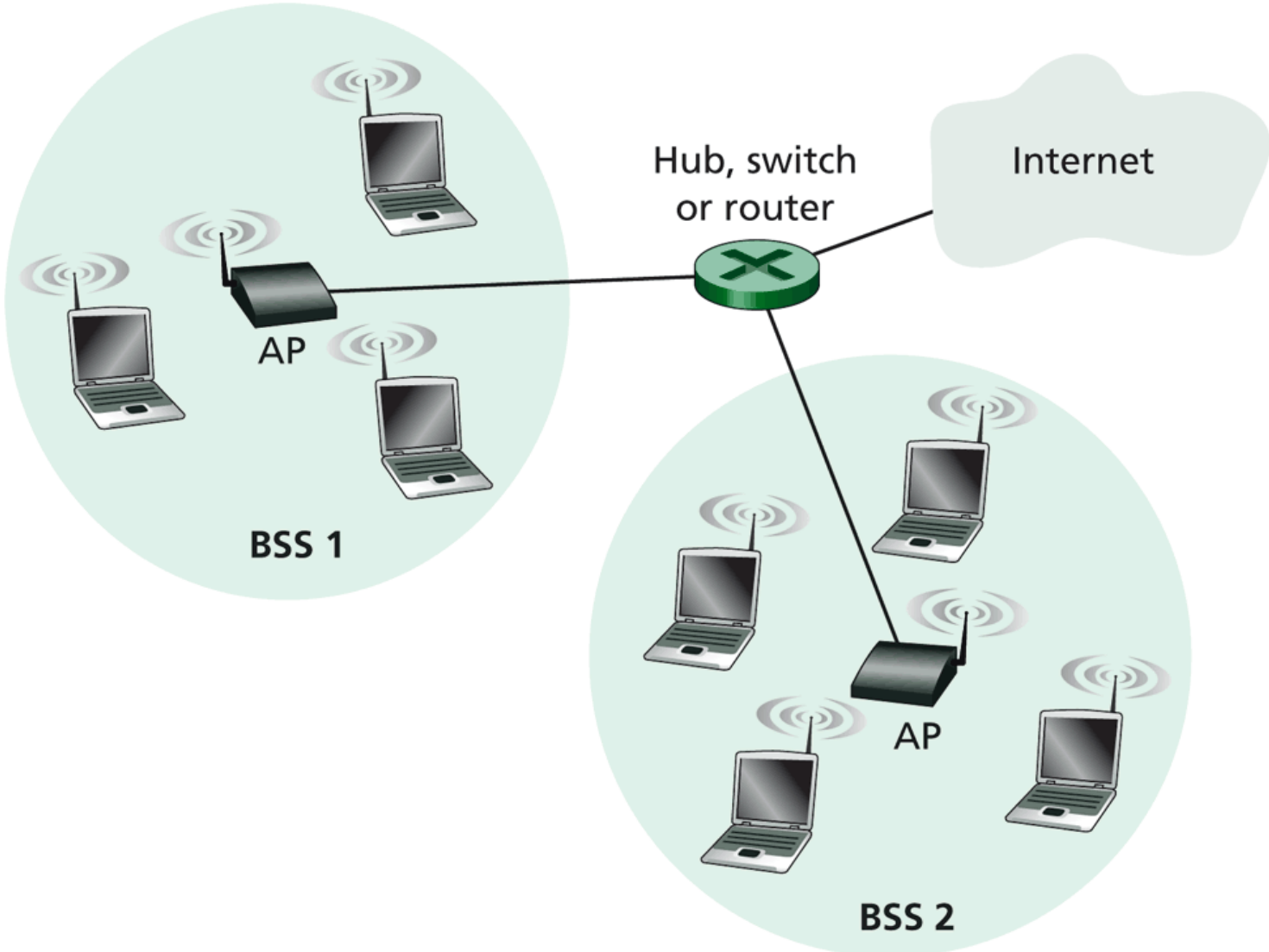


Figure 6.6 ♦ IEEE 802.11 LAN architecture

28.3.2. Modo Ad-Hoc

- Las estaciones se comunican directamente entre porque están lo suficientemente próximas.

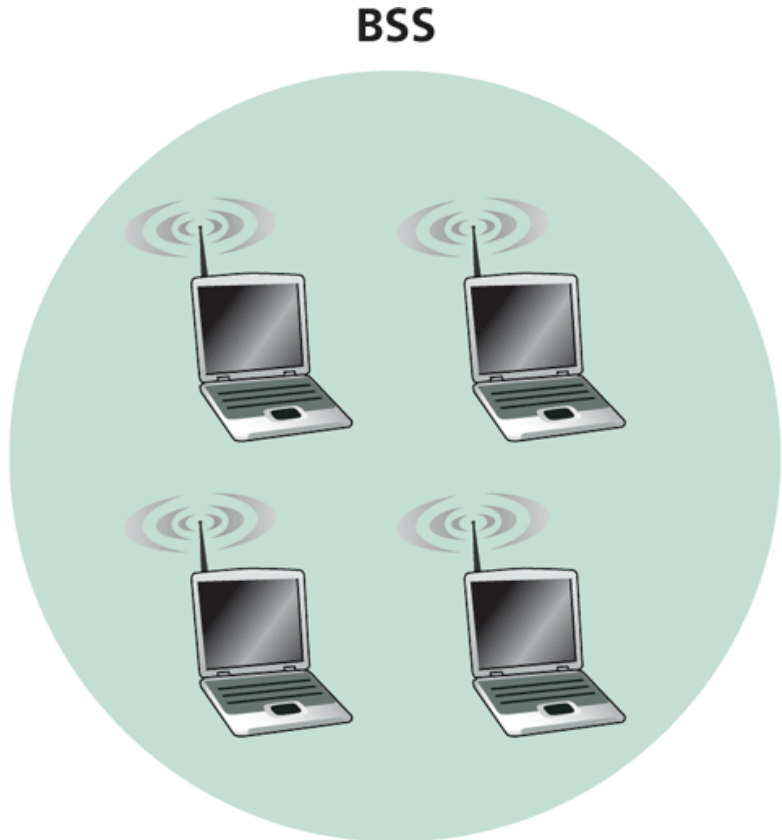


Figure 6.7 ♦ An IEEE 802.11 ad hoc network

28.4. Canales

- Con la idea de acomodar a más de un AP en una misma región geográfica, el 802.11 define un conjunto de canales de frecuencia.
- Existen un total de 11 canales que está parcialmente solapados entre sí.
- Sólo los canales 1, 6 y 11 no se solapan entre sí.

28.5. El proceso de asociación

- Los AP's generan periódicamente *beacon frames* con la idea de anunciar su presencia a los nodos móviles.
- En un *beacon frame* figura la dirección física y la SSID (Service Set Identifier) del AP. La SSID es una cadena alfanumérica que asigna el administrador de la red al AP.
- Los *beacon frames* nunca están cifrados.
- Los *beacon frames* sólo se transmiten a través del canal para el que se ha configurado la emisión del AP.

- Cuando un nodo móvil se intenta asociar a un AP pueden ocurrir dos cosas:

  1. Que el AP no controle quién se asocia.
  2. Que el AP sí lo haga. En este caso el control se realiza de dos formas distintas:

     a) Permitiendo sólo la conexión desde las direcciones físicas previamente especificadas.

     b) Permitiendo sólo la conexión desde aquellos nodos que sean capaces de autentificarse.[1]

[1] Usando protocolos como RADIUS o DIAMETER [15]. Esto no tiene nada que ver con el cifrado de datos de tipo WEP o WPA.

28.6. CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance)

- Wi-Fi es semejante a una red Ethernet de medio compartido. Sin embargo, el protocolo CSMA/CD no es suficiente a causa del *problema del terminal oculto* que puede provocar que dos nodos provoquen colisiones sin ent

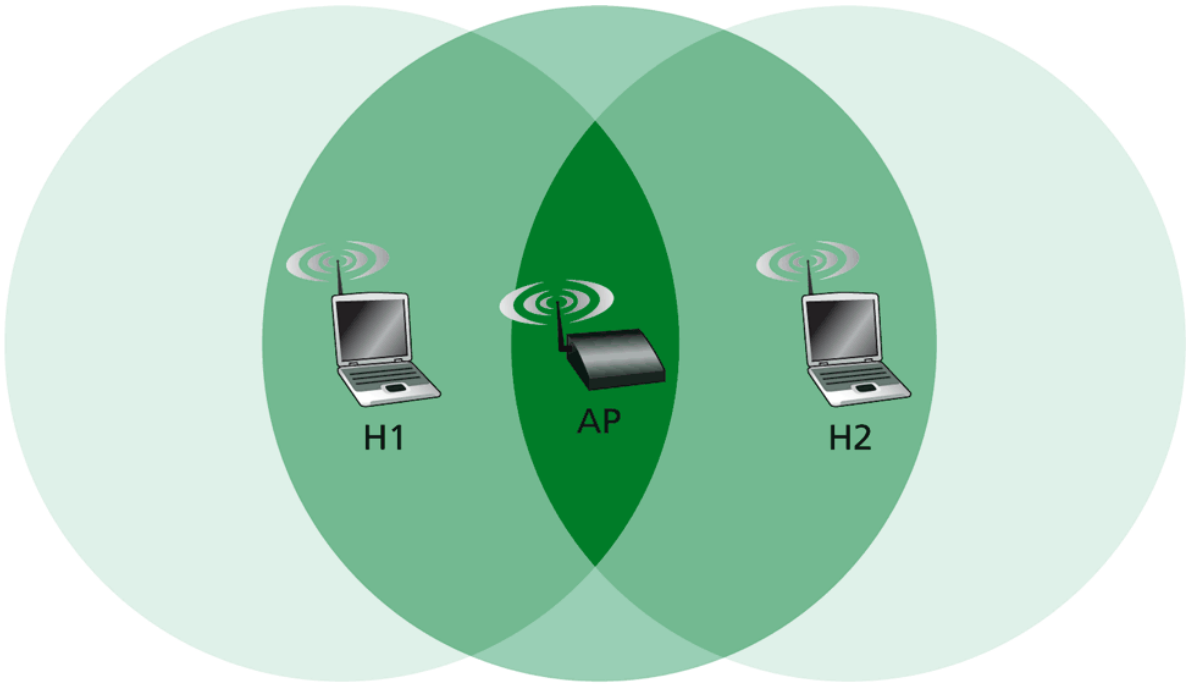


Figure 6.9 ♦ Hidden terminal example: H1 is hidden from H2, and vice versa

- Este problema provoca las siguientes diferencias:

  1. Las estaciones transmiten siempre frames completos (2346 bytes), y por tanto, nunca abortan la transmisión porque no hay forma de comprobar si se ha producido una colisión testeando simplemente si lo que se inyecta en el enlace es lo que se lee del mismo.
  2. Para conocer si la transmisión de un frame de datos ha tenido éxito se espera la llegada de un frame ACK con un tiempo máximo de espera. Si éste no llega a tiempo se retransmite el frame de datos y el proceso se repite.[2]

[2] Y esto se hace a nivel de la capa de enlace de datos.

DIFS = Distributed Inter-Frame Space (chequeo de portadora).

SIFS = Short Inter-Frame Spacing.

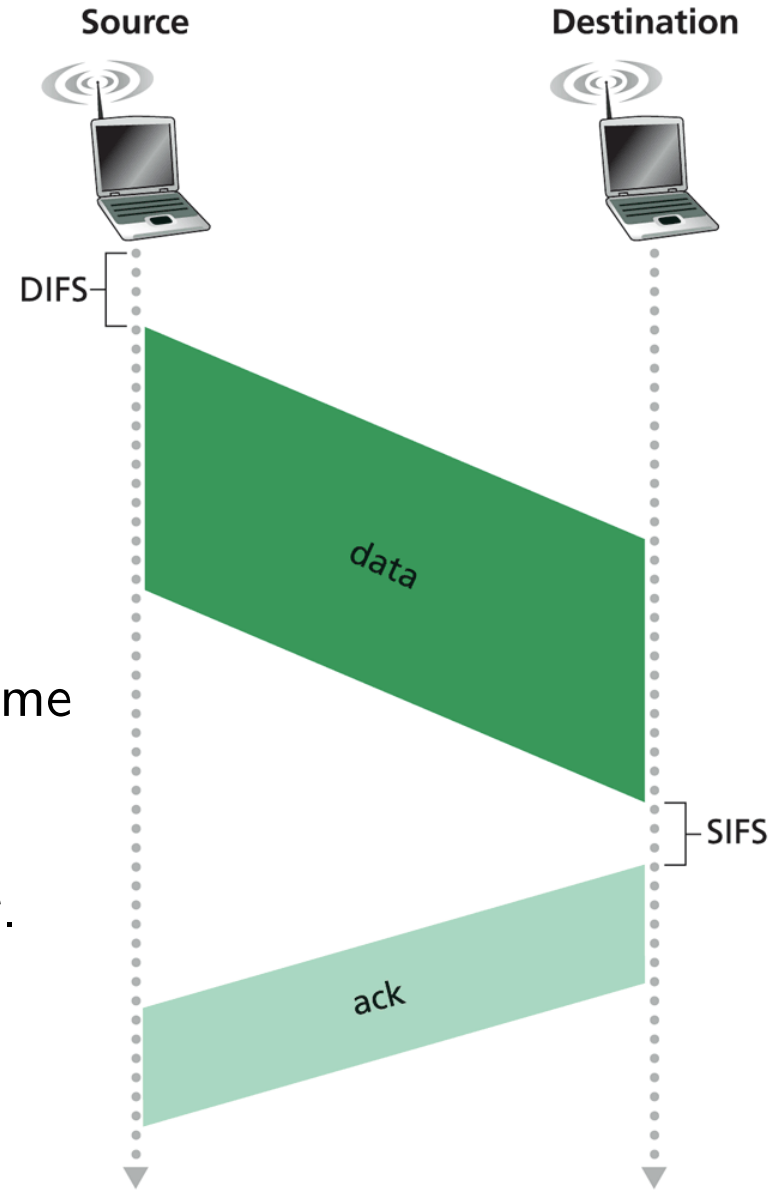


Figure 6.8 ♦ 802.11 uses link-layer acknowledgments

- Las anteriores propuestas no evitan las colisiones porque los nodos pueden acceder al medio para transmitir al mismo tiempo. Cuando los frames son pequeños esto no es un gran problema.
- Sin embargo, cuando los frames son largos las colisiones desperdician mucho ancho de banda (recuérdese que se transmiten siempre frames completos). En este caso los nodos *pueden* usar un sistema de reserva de ancho de banda llamado RTS/CTS.
- Cuando un nodo genera un frame RTS (Request To Send) es que va a enviar un gran frame de datos y solicita al nodo receptor que genere un frame CTS.
- Un frame CTS (Clear To Send) indica a todos los nodos que lo reciben (excepto al que ha enviado el RTS) que deben esperar a que el medio quede libre (alrrededor del nodo receptor) antes de transmtir. Nótese que como la potencia de transmisión de todos los nodos debe ser aproximadamente la misma, el frame CTS llegará potencialmente a todos los nodos que podrían colisionar en las inmediaciones del nodo receptor (el que envía el CTS).

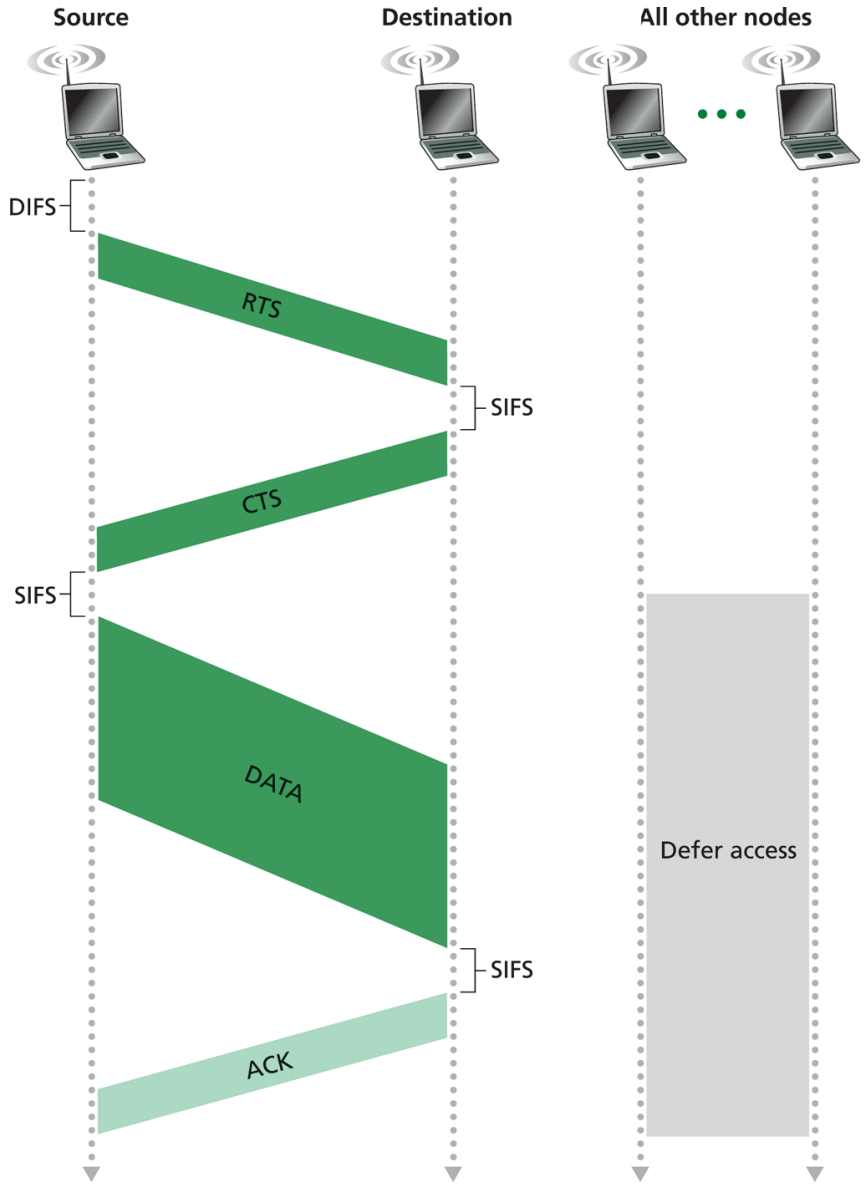


Figure 6.10 ♦ Collision avoidance using the RTS and CTS frames

28.7. Estructura del frame IEEE 802.11

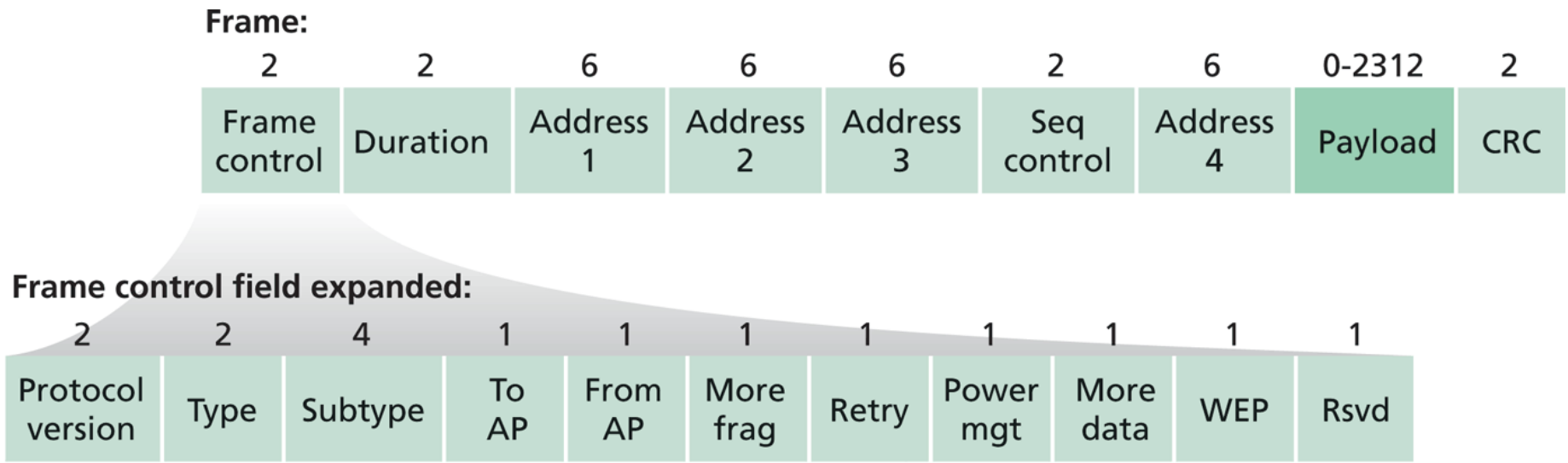


Figure 6.11 ♦ The 802.11 frame

- **Frame control:** Diversa información sobre la transmisión del frame.
- **Duration:** Usado en el frame RTS. Sirve para reservar ancho de banda durante un determinado intervalo de tiempo.
- **Address 1:** Dirección física del interface de red Wi-Fi origen.
- **Address 2:** Dirección física del interface de red Wi-Fi destino.

- **Address 3:** Dirección física del interface de red Ethernet del router (si el frame va dirigido hacia el router de la subred). Sin este campo sería imposible enviar mensajes fuera de la BSS (ver Figura 6.12 de [15]. Para enviar un frame desde un host móvil hacia Internet, Address 1 tendría la MAC del host origen, Address 2 la MAC del correspondiente AP y Address 3, la MAC del router).
- **Seq control:** Número de secuencia usado en el protocolo ARQ de parada y espera para controlar las colisiones.
- **Address 4:** NPI. ¿Tal vez para enviar a otro BSS?
- **Payload:** Los datos (algo más 2K bytes a lo sumo).
- **CRC:** Un CRC-16.

28.8. Mobility

- Los nodos móviles pueden cambiar con facilidad de BSS:

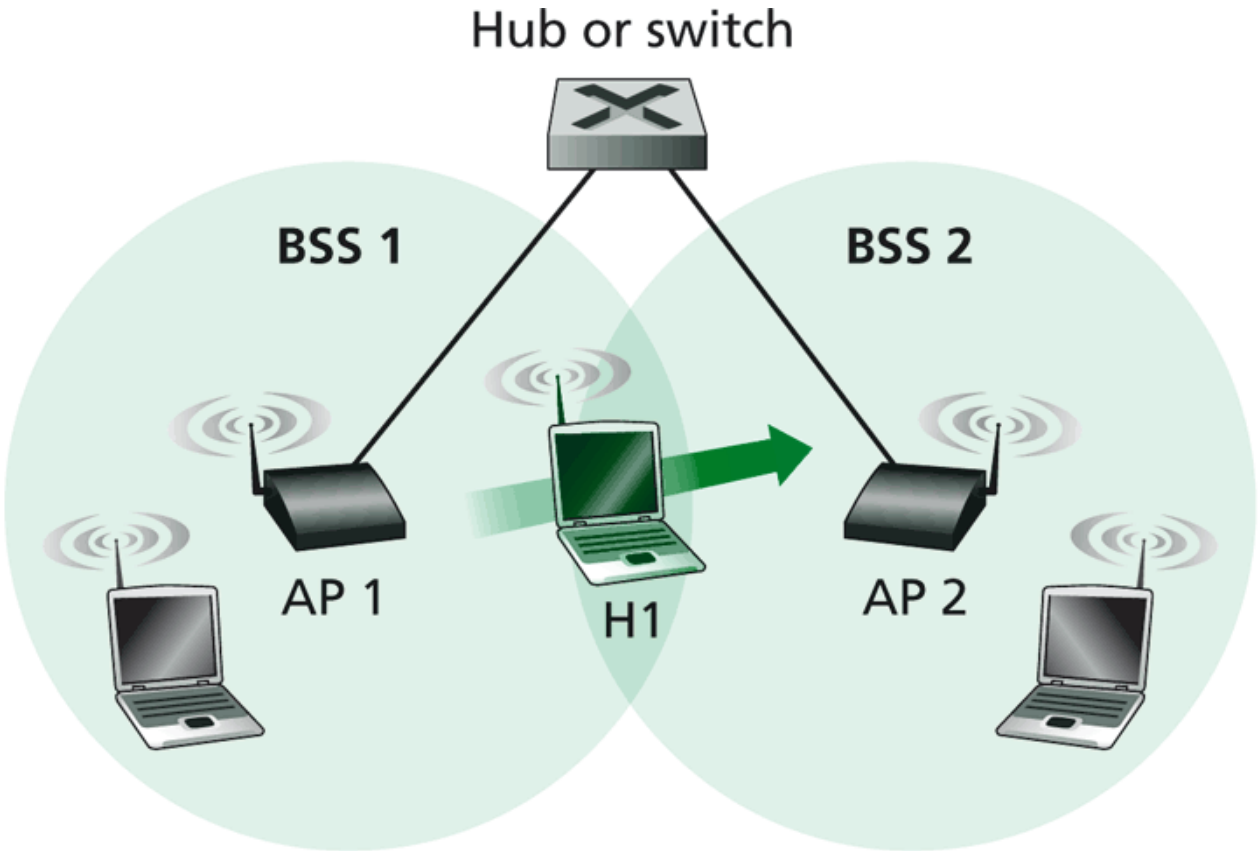


Figure 6.13 ♦ Mobility in the same subnet

- Cuando se usa un hub para conectar las BSS no existen problemas para entregar los frames al nodo móvil. Sin embargo, cuando hay un conmutador éste debe actualizar su *switch table* lo más frecuentemente posible (en cuanto la estación se asocia al nuevo AP).

Capítulo 29

Bluetooth

29.1. IEEE 802.15

- Tasa de trasmisióm modesta (hasta 721 Kbps) [14].
- Misma banda de frecuencias que 802.11 ($2{,}45$ GHz - $2{,}5$ GHz).
- Tecnología pensada para reemplazar los cables en distancias cortas (impresoras, ratones, teclados, ...), hasta un máximo de 10 metros.
- Incorpora control de flujo y de errores usando un protocolo parada y espera.
- El protocolo MAC está basado en un protocolo basado en turnos mediante sondeo.

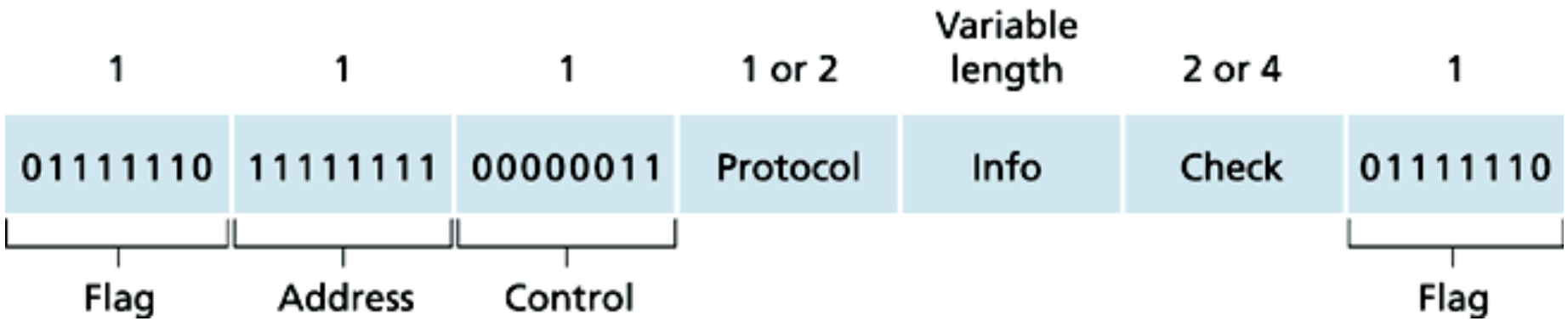
Capítulo 30

Enlaces PPP

30.1. EL PPP (Point-to-Point Protocol)

- RFC's 1661, 1547 y 2153.
- El PPP se utiliza en enlaces punto-a-punto [14]. Ejemplo: las conexiones a través de modem al ISP.
- Principales características:
  - **Packet framing**. El comienzo y el fin de cada frame es señalizado.
  - **Independencia del protocolo usado en la capa de red**. Permite usar los protocolos IP, DECnet, AppleTalk, etc.
  - **Independencia de la capa física utilizada**. PPP se puede utilizar en enlaces serie o paralelos, conexiones telefónicas analógicas, ISDN, ADSL, SONET/SDH, X.25, etc.
  - **Detección de errores de transmisión**. Sólo los detecta, no los corrige.
  - **Detección de fallo en el enlace de transmisión**.

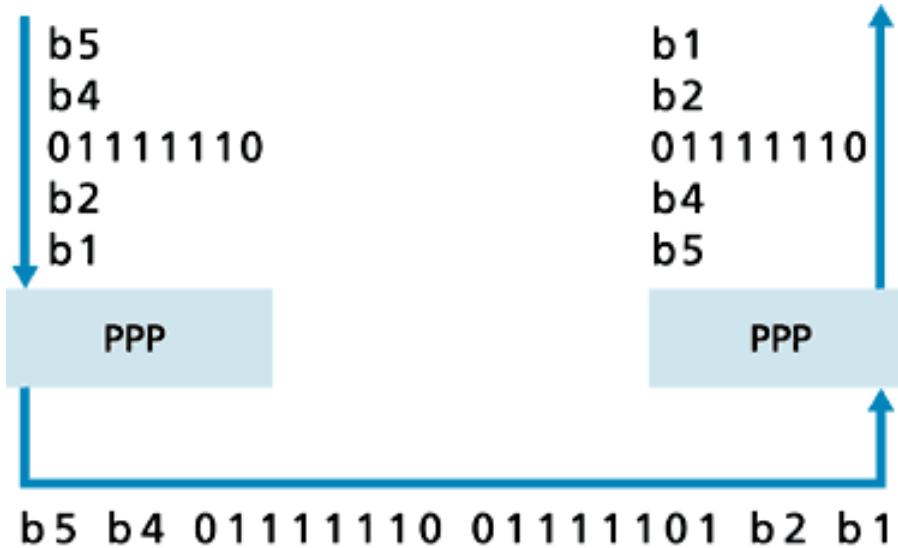
30.2. Data framing



- **Flag**. El comienzo y el fin del frame se indica con la secuencia 0111 1110.
- **Address**. Siempre igual a 1111 1111 (genial).
- **Control**. Siempre igual a 0000 0011 (también genial).
- **Protocol**. Protocolo usado en la capa superior.
- **Information**. Datos transmitidos.
- **Check(sum)**. Se utiliza un CRC.

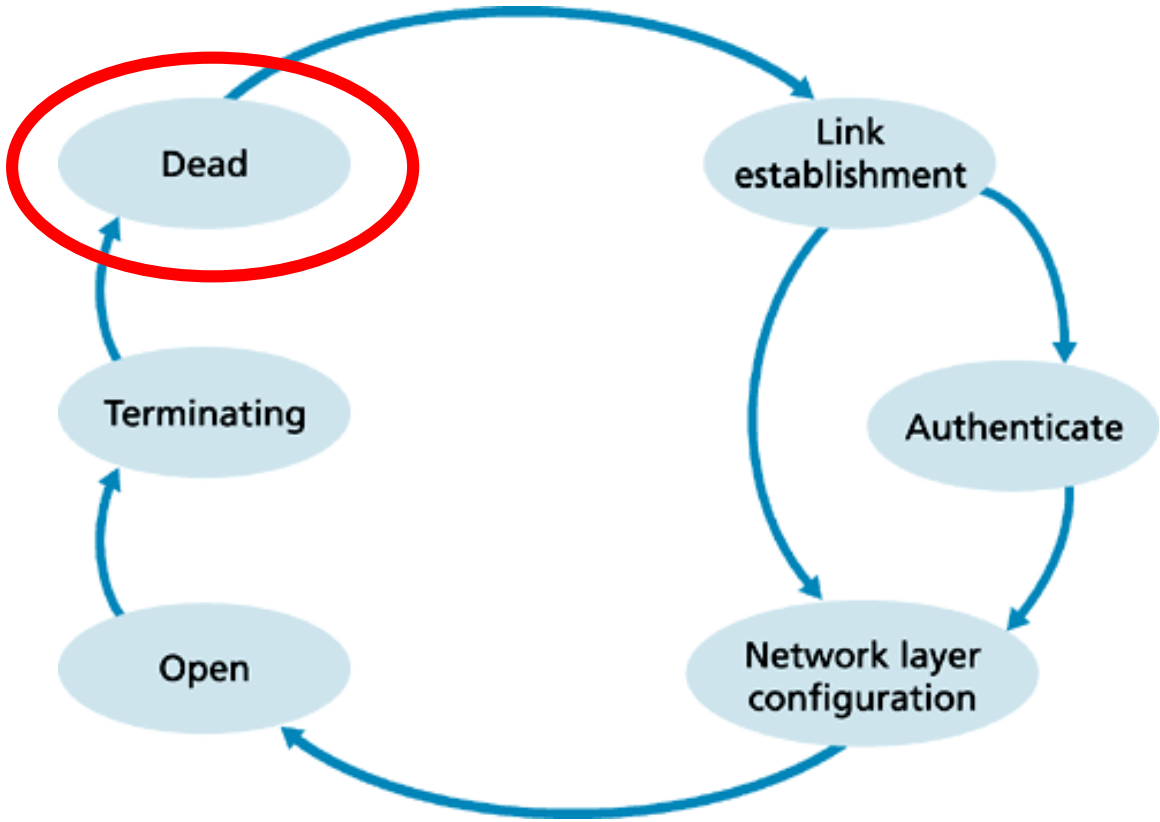
30.2.1. Byte stuffing

- Evita que cuando el patrón 0111 1110 aparezca en el campo de información del frame no se produzca un error de finalización prematura del frame.
- Cuando este byte aparece en los datos, el PPP lo antecede del byte de control 0111 1101 para indicar que no se trata del fin de frame.
- Si el byte 0111 1101 aparece en los datos, el PPP lo antecede de otro byte de control 0111 1101 para indicar este evento. Por este motivo a este byte se le llama también *código de escape*.

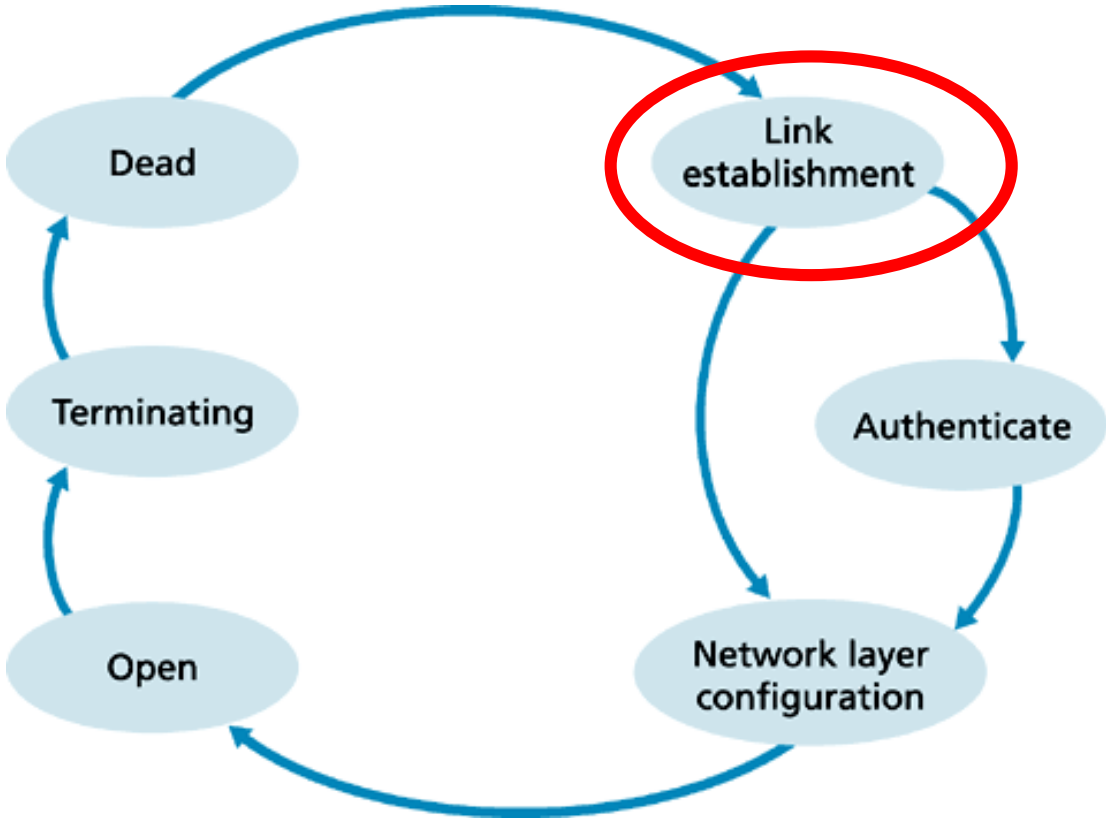


30.3. Protocolo de control del enlace

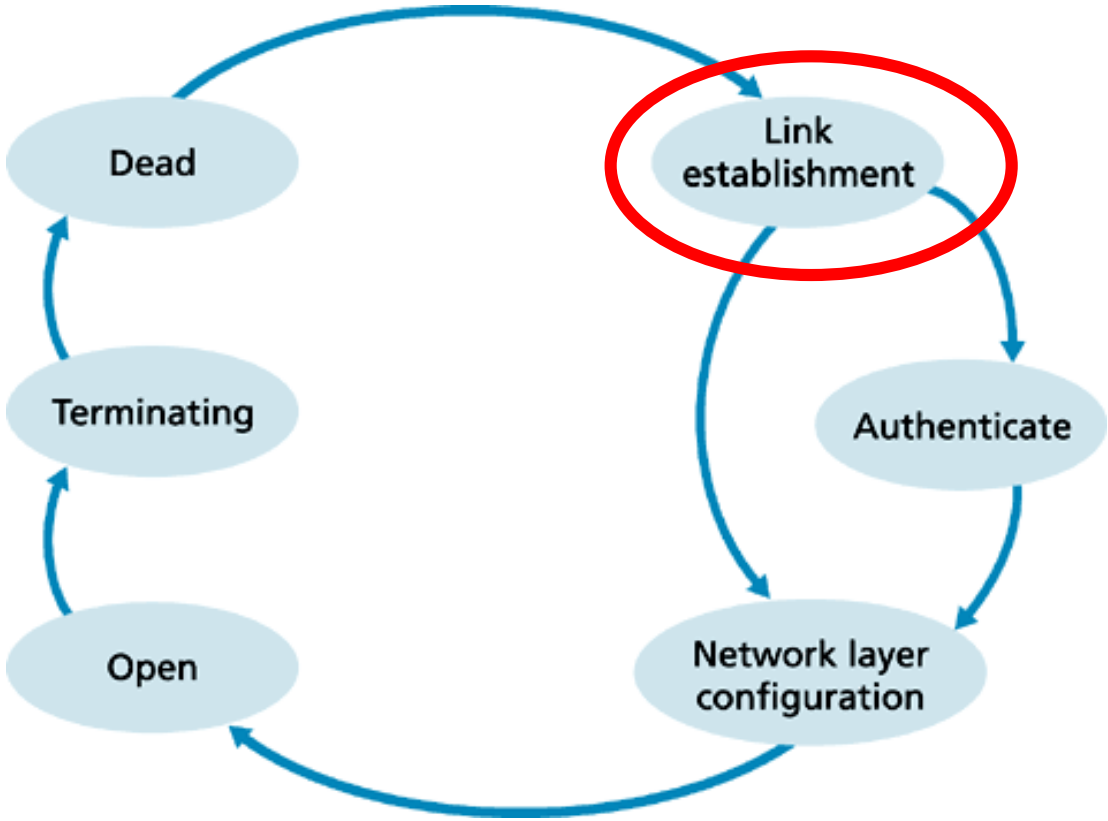
- Una conexión PPP siempre comienza y termina en el estado "Dead".



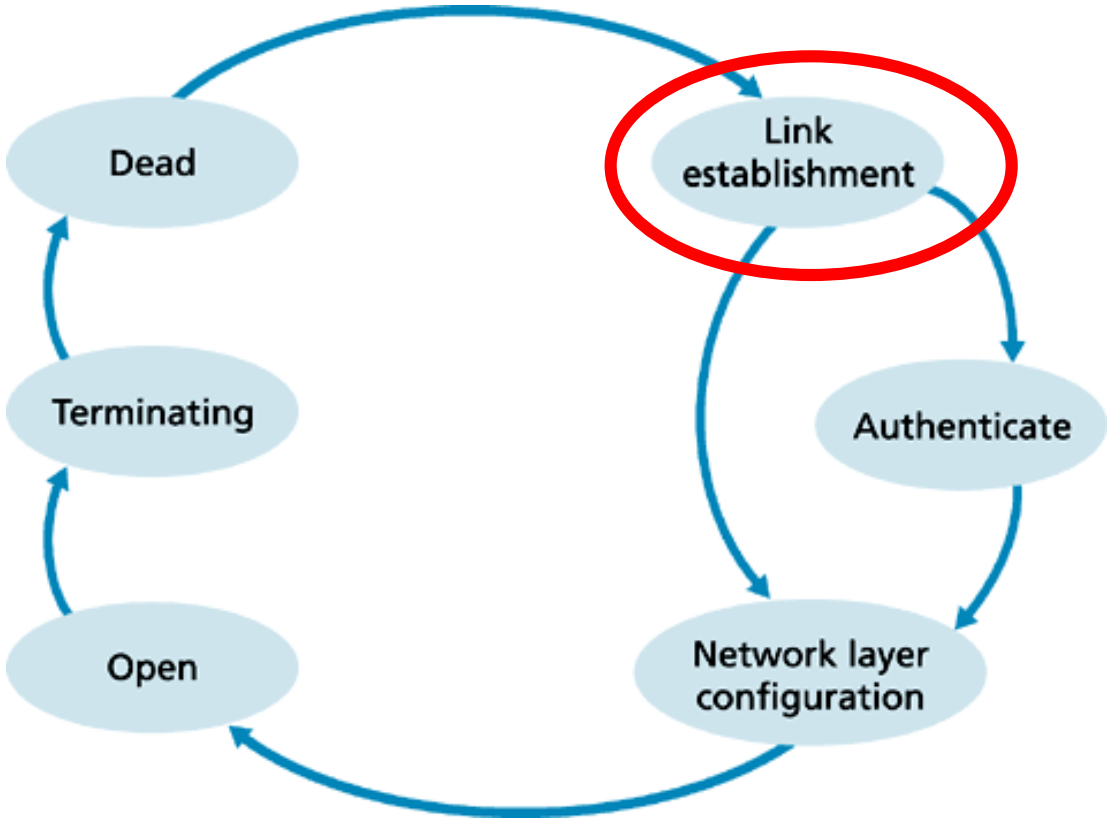
- Antes de transmitir o recibir se entra en el estado "Link establishment".



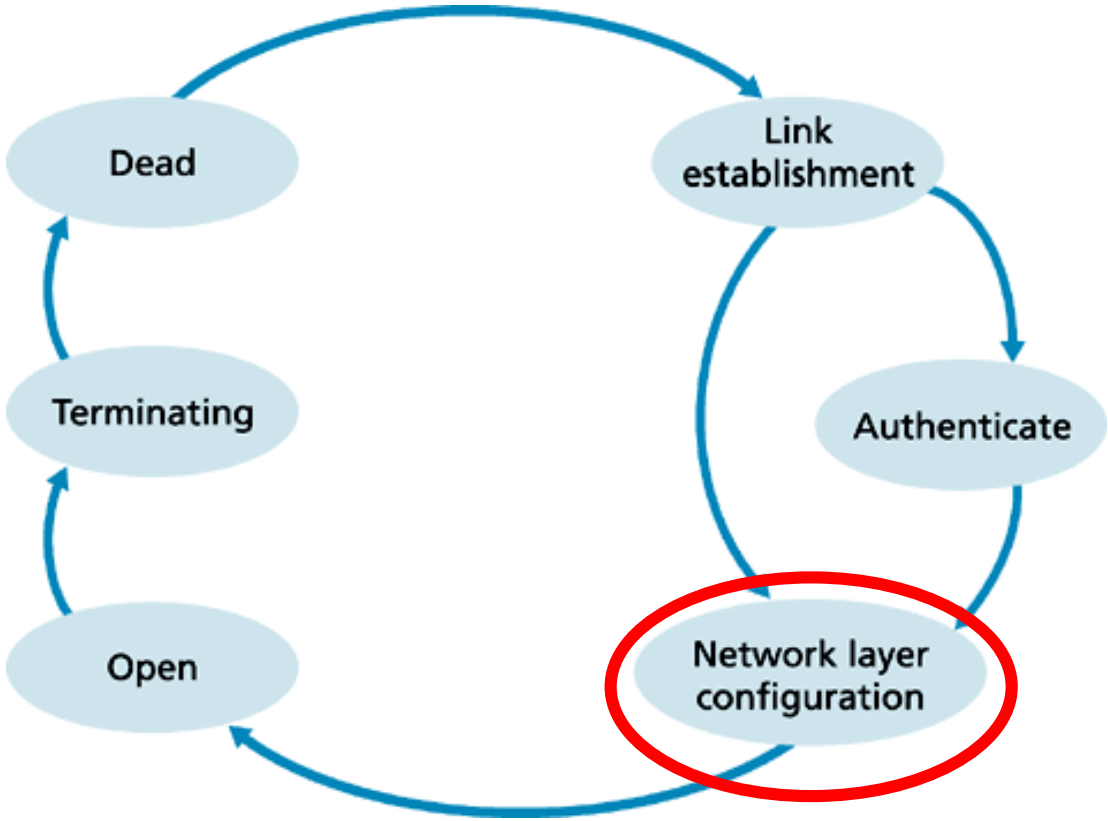
- El lado que trata de establecer la conexión envía al otro lado sus "preferencias" de configuración del enlace (tamaño máximo de frame, protocolo de autentificación, etc.).



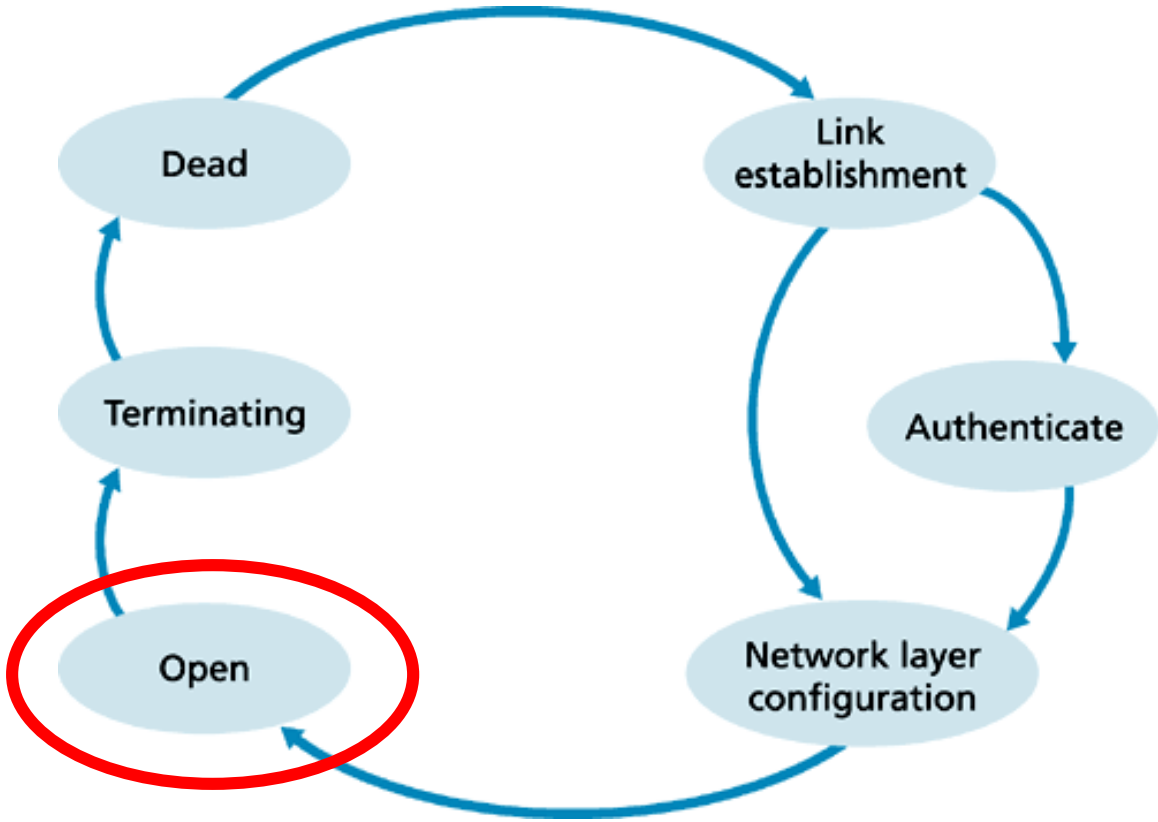
- El lado "servidor" responde indicando qué prefrerencias son aceptables (incluyento todas o ninguna, en cuyo caso no se establece la conexión PPP). Opcionalmente solicita una identificación.



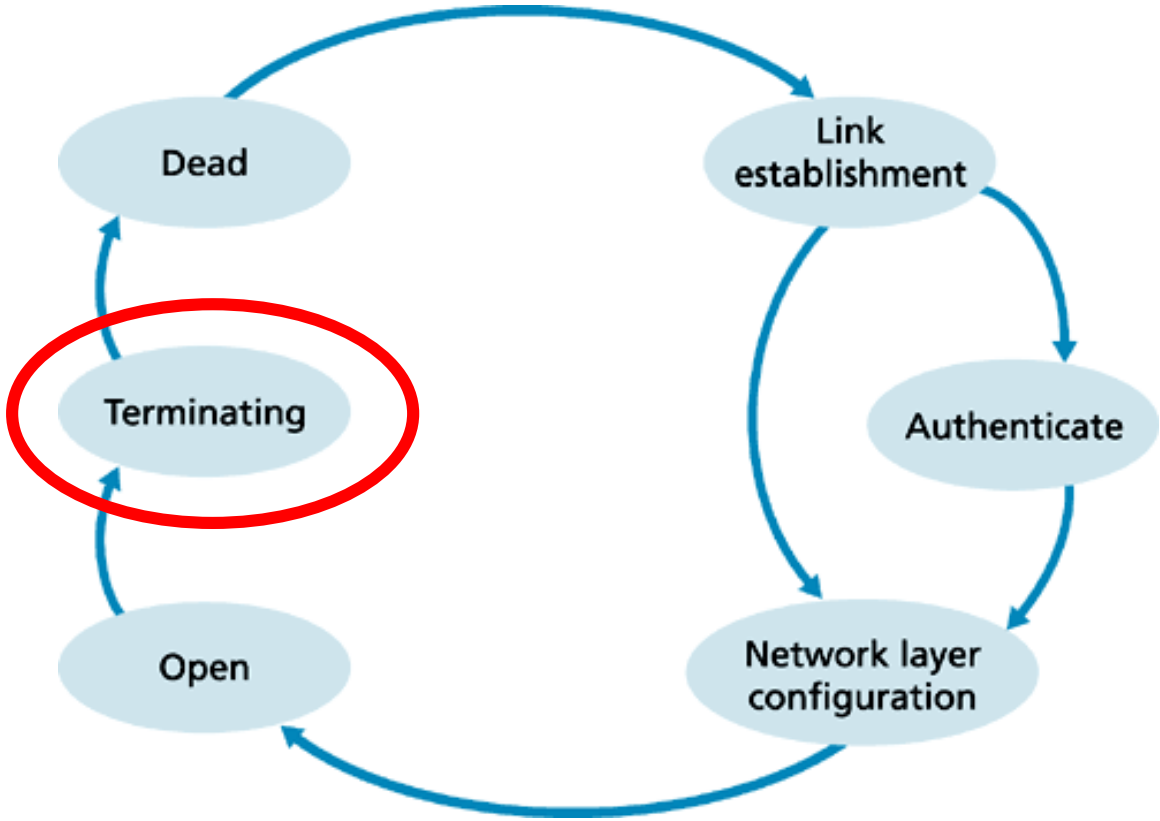
- En ese instante intercambian mensajes las capas de red de ambos extremos. Por ejemplo, cuando se utiliza el IP se configuran las dirs IP, el formato de compresión de los datos, etc.



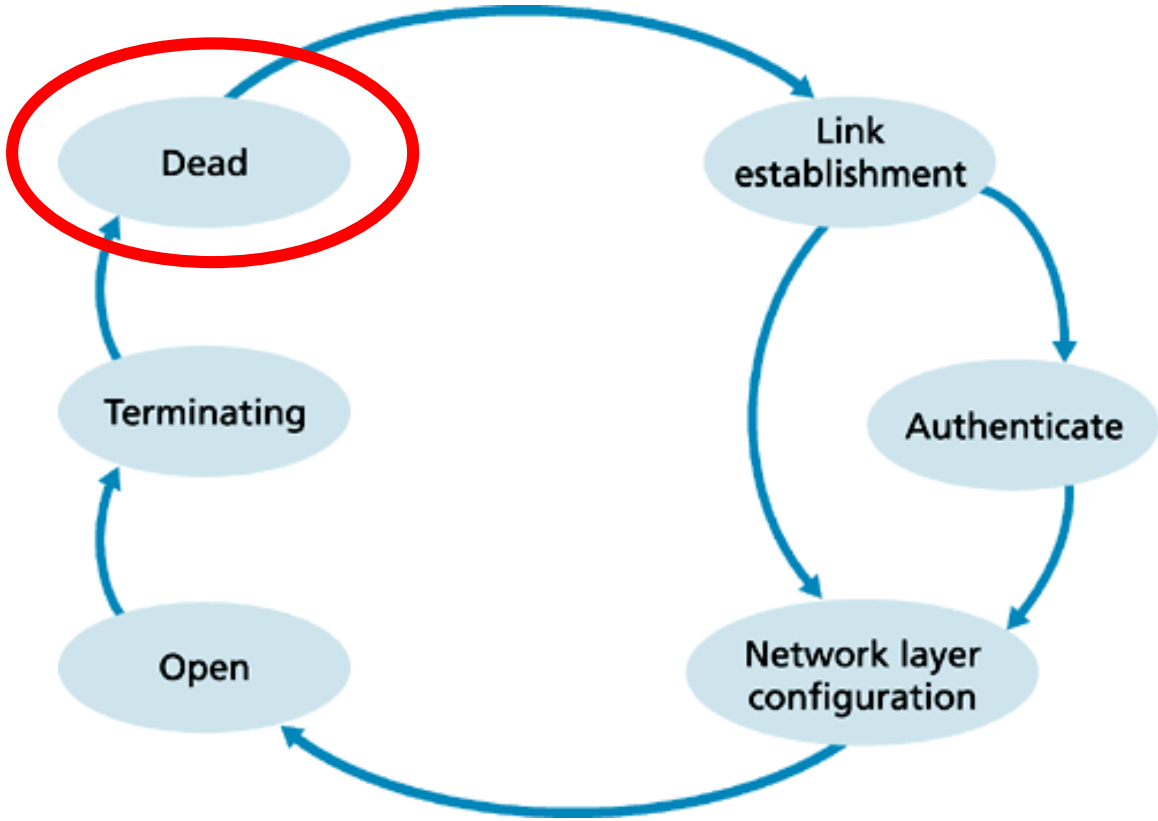
- En el estado "Open" se transmiten los datos, frame a frame. El tamaño de estos viene determinado por el IP.



- La conexión PPP permanece abierta hasta que uno de los extremos envía un mensaje de solicitud de terminación. Este debe ser confirmado por el otro extremo.



- Cuando los dos extremos cierran la conexión y sitúan, de nuevo, en el estado "Dead".



Capítulo 31

ATM

31.1. Historia

- ATM fue desarrollada por el foro ATM y la ITU a mediados de los 80 [7]. En aquella época era impensable la transmisión en tiempo real de flujos multimedia (principalmente voz) usando redes de conmutación de paquetes [14].
- ATM proporciona una solución completa para las aplicaciones de red (implementa todas las capas excepto la de aplicación).
- Por desgracia (para ATM) cuando estuvo realmente operativa la inmensa mayoría de las aplicaciones de red (el WWW, teléfono IP, el correo electrónico, etc.) ya estaban funcionando sobre TCP/IP.
- En este sentido, y aunque ATM podría haber sustituido al TCP/IP, actualmente se utiliza fundamentalmente para transmitir paquetes IP. Por este motivo en esta exposición se ha considerado a ATM como una tecnología más a nivel de enlace de datos.

31.2. Principales características

- **Altas tasas de transmisión**. ATM funciona prácticamente sobre cualquier capa física. En los enlaces de larga distancia suele utilizarse fibra óptica. Velocidades típicas de transmisión (utilizando SONET) son $155{,}52$ Mbps y 622 Mbps.
- **Enlaces de larga distancia**. Los enlaces ATM son punto a punto y pueden recorrer mucha distancia (miles de Km).
- **Routing a nivel de enlace de datos**. Los conmutadores ATM incorporan mecanismos de routing a nivel de ATM muy semejantes a los utilizados por el IP.
- **Calidad de servicio**. ATM proporciona mecanismos de control de la congestión y de reserva de ancho de banda. Esto permite que el usuario que paga puede obtener una calidad de servicio superior.

31.3. Modelo de servicio

- ATM proporciona varios modelos de servicio que se adaptan a diferentes necesidades [32]:
  - **CBR (Constant Bit Rate)**. Pretende simular un enlace dedicado. Los bits se ponen en un extremo y un cierto tiempo después, siempre fijo, salen por el otro (jitter igual a 0). No existe control de flujo entre el emisor y la red. Aplicación típica: transportar conversaciones telefónicas.
  - **VBR (Variable Bit Rate)**. La tasa de bits varía con el tiempo y las latencias también. Sin embargo, pueden negociarse latencias máximas cuando usamos la modalidad RT-VBR (Real Time VBR). La red no proporciona control de flujo. Aplicación típica: vídeo MPEG.
  - **ABR (Available Bit Rate)**. Es un caso intermedio entre CBR y VBR. La red proporciona una tasa de transmisión mínima, aunque si la carga no es alta, entonces pueden alcanzarse tasas

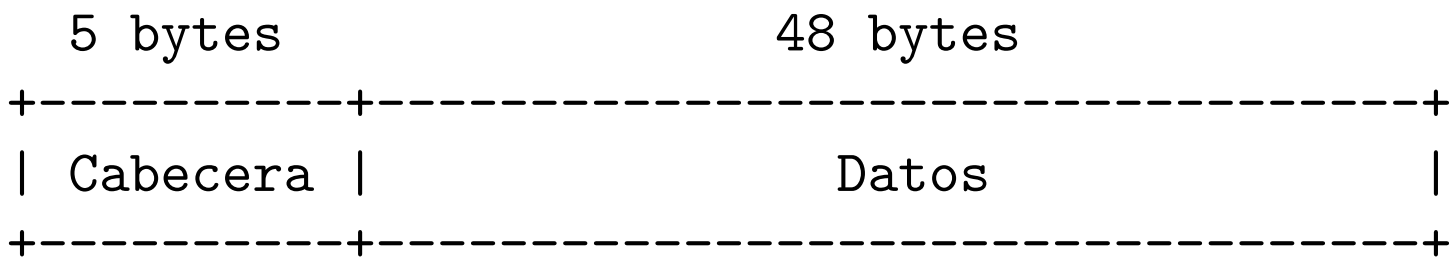
mayores. La red proporciona control de congestión. Ejemplo: contrato para un número pico de llamadas telefónicas.

- **UBR (Unspecified Bit Rate)**. Igual que la anterior, pero la red no proporciona control de congestión. Esto es útil cuando este proceso se realiza en una capa superior (TCP/IP). Ejemplo: transferencia de ficheros, WWW, e-mail.

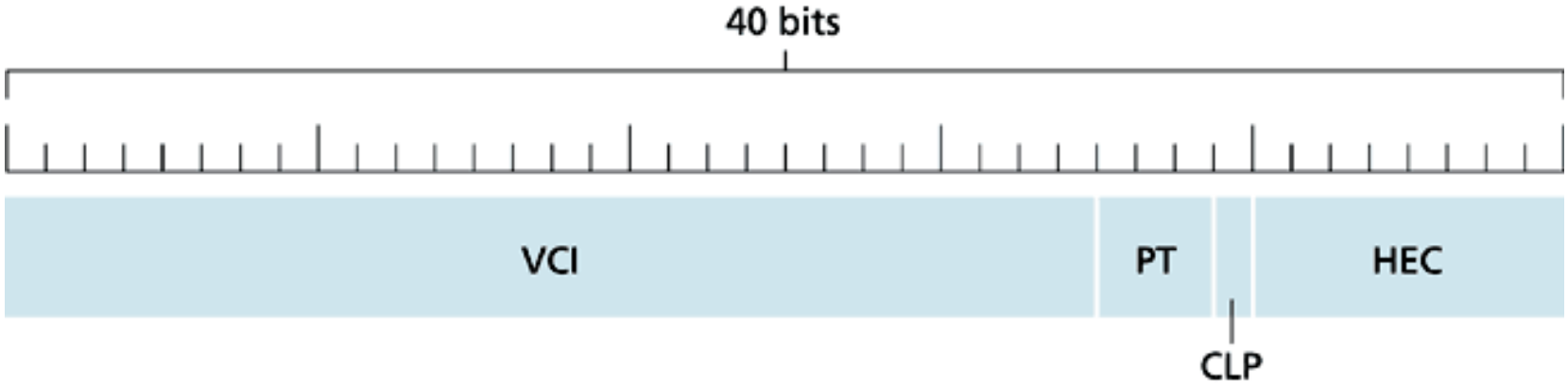
- En ningún caso los errores de transmisión tratan de corregirse mediante retransmisión. Sin embargo se utiliza un código de detección y corrección de errores en la cabecera. Si esta se corrompe no se retransmite la celda.

31.4. Las celdas ATM

- Se utiliza TDM asíncrona (como los routers).
- Los conmutadores ATM encaminan celdas de longitud fija (5 bytes de cabecera y 48 bytes de datos).



- El formato de la cabecera es:



  - **VCI (Virtual Channel Indentifier)**. Establece el VC que la celda debe seguir.
  - **PT (Payload Type)**. Indica el tipo de carga útil que transporta la celda: (1) datos, (2) información de control o (3) celda de relleno (cuando el medio no permite dejar de transmitir celdas).
  - **CLP (Cell-Loss Priority)**. Identifica la prioridad de la celda a la hora de ser desechada (usada en el mecanismo de control de congestión de los conmutadores).
  - **HEC (Header Error Control)**. Código de Hamming que permite detectar y corregir errores (en la cabecera).

31.5. Morfología de la red

- Idéntica a la de una red TCP/IP donde los routers son conmutadores ATM y los enlaces son siempre punto a punto (no pueden ser redes).
- Los routers IP ven la red ATM como una tecnología más de interconexión (como Ethernet). Se conectan a la red ATM a través de un adaptador ATM.
- Los nodos también pueden conectarse directamente a un conmutador ATM aunque necesitan un adaptador ATM.

31.6. Canales virtuales y routing

- ATM utiliza circuitos (canales en la jerga ATM) virtuales (VC's). Estos pueden ser permanentes o temporales.
- Las celdas nunca llegan desordenadas al receptor.
- Los VC's temporales implican un establecimiento previo de conexión a la transmisión útil de celdas.
- En la mayoría de los enlaces de larga distancia los canales son permanentes (routing estático) y por lo tanto el establecimiento de conexión no existe en estos casos.
- El estándar ATM no define ningún algoritmo de routing concreto. Se deja en manos del constructor del conmutador dicha elección [32]. De todas formas, dado el uso que tiene actualmente ATM (enlaces de larga distancia), el routing en ATM en la mayoría de los casos se realiza de forma estática.

31.7. Arquitectura de ATM

ATM se organiza en 4 capas:

Capa de Usuario
Capa AAL
Capa ATM
Capa Física

31.7.1. La capa física

- Controla los voltajes, relojes maestros, framing, ... es decir, se encarga de la transmisión física de las celdas ATM.
- Realiza en control de errores (descarta las celdas con cabeceras erróneas).[1]
- La capa física se divide en 2 sub-capas:
  1. **PMD (Physical Medium Dependent) sub-layer**. Diferente para tipo de medio físico que es capaz de transmitir ATM.
  2. **TC (Transmission Convergence) sub-layer**. Dependiente de la capa física. Se encarga fundamentalmente del mantenimiento del sincronismo de los relojes del emisor y del receptor y de chequear los errores en las cabeceras.

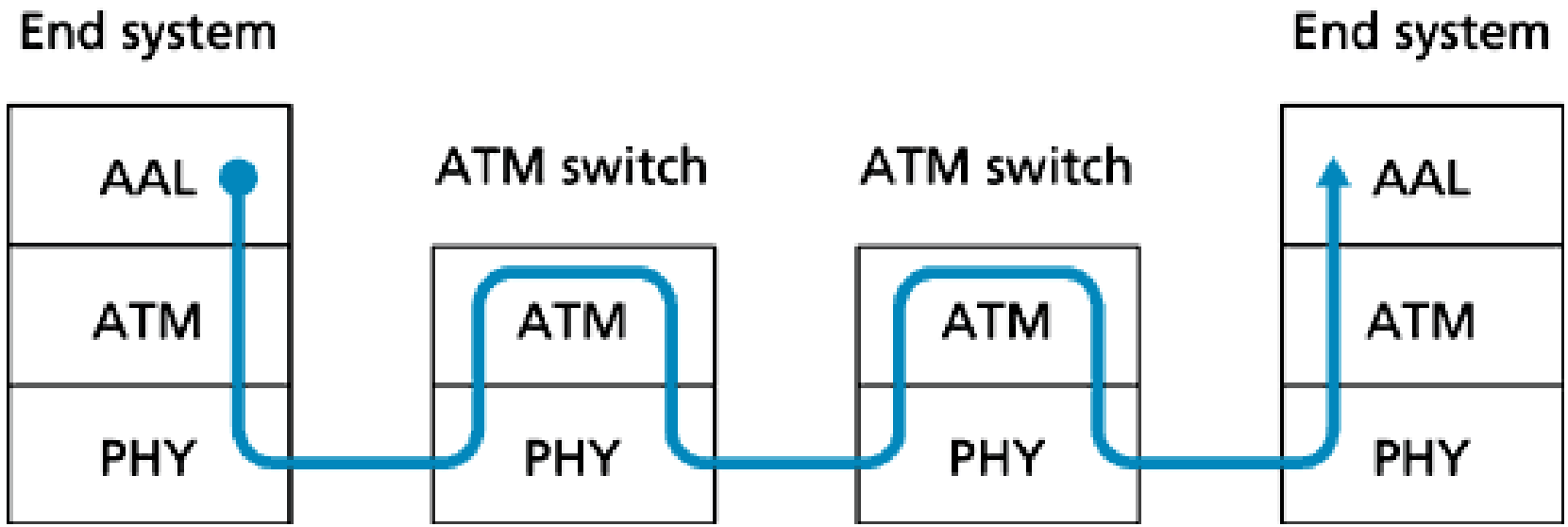
[1] Se utiliza un código de Hamming que es capaz de corregir un error y detectar errores múltiples.

31.7.2. La capa ATM

- Establece las conexiones [7].
- Realiza el framing (segmentación de los datos generados por la capa superior en celdas).
- Control de la congestión (sólo en el modo ABR).

31.7.3. La capa AAL (ATM Adaptation Layer)

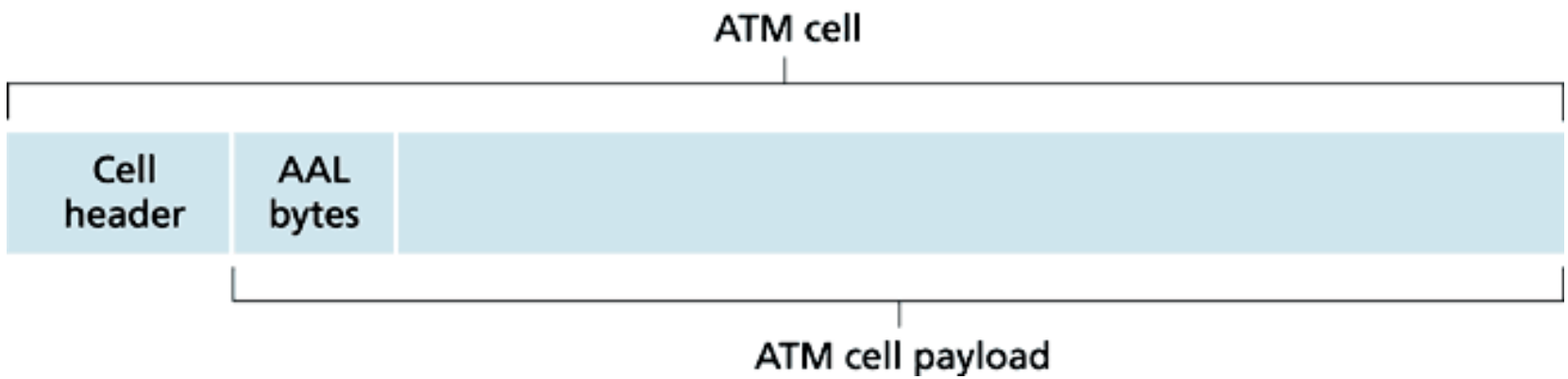
- Permite que los protocolos de la capa de red (IP en la mayoría de los casos) puedan usar ATM.
- AAL se implementa sólo en los extremos (hosts y routers) de una red ATM.



- La AAL puede ser considerada como "la capa de transporte de ATM". Existen 3 AAL's diferentes dependiendo del tipo de servicio que se va a prestar:

Protocolo	Uso
AAL 1	CBR (telefonía)
AAL 2	VBR (audio y vídeo comprimido)
AAL 5	UBR (datagramas IP)

- La capa AAL genera su propia información de control y puede "robar" espacio a los datos útiles.



- En el caso de la AAL 5, la AAL no introduce ninguna información de control. Sin embargo cuando hasta la AAL 5 llega un datagrama IP se transmite la siguiente información:

donde

  - **CPCD-PDU payload**. Sección de un datagrama IP.
  - **PAD**. Relleno de bytes para que el número de bytes de datos transmitidos sea un múltiplo de 48.
  - **Length**. Número de bytes de datos transmitidos.
  - **CRC**. Código de redundancia cíclica (mismo que Ethernet).

31.7.4. La capa de usuario

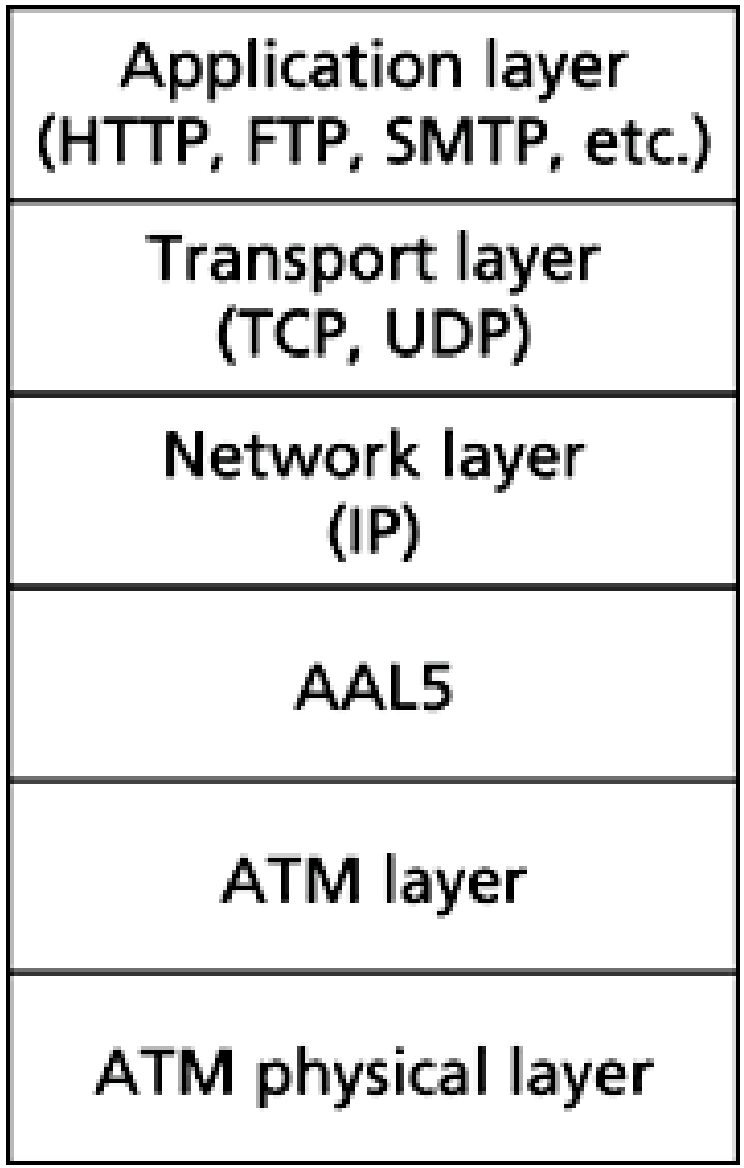
- En ella se ejecutan las aplicaciones de red.
- Realiza la corrección de errores y de flujo cuando es necesario.

31.8. Control de la congestión

- ATM realiza un control de la congestión sólo en su modo ABR. En este modo los conmutadores pueden desechar celdas cuando se congestionan con el objetivo de mantener las tasas y las latencias contratadas.
- En una transmisión ATM, entre las celdas de datos el emisor inserta celdas de control llamadas celdas RM (Resource Management).[2]
- Cuando las celdas de control atraviesan los conmutadores, estos modifican unos determinados campos de ellas cuando comienzan a estar congestionados.
- Cuando las celdas alcanzan al receptor este las devuelve al emisor (posiblemente modificándolas si se encuentra congestionado). El receptor también puede generar celdas RM si estas no le llegan.
- El emisor regula su tasa de transmisión en función de la información de las celdas RM retornadas.

[2] Típicamente 1 de cada 32 es una celda de control.

31.9. The Internet-over-ATM protocol stack



Apéndices

Apéndice A

Espectro de una señal digital

A.1. Dato e información

- Estos dos términos pueden parecer sinónimos. Sin embargo, no es así: *Los datos son la representación física de la información*.
- ¿Qué es la *información*? *La información es todo aquello que una vez conocido reduce la incertidumbre de algo*. Por ejemplo, el resultado de la próxima quiniela de futbol es la información que transportaría un boleto de quiniela premiado para la próxima jornada futbolística.

A.2. Tipos de fuentes de información

- Desde el punto de vista discreto o continuo de la información que generan, existen dos tipos de fuentes de información, las digitales y las analógicas. La diferencia radica en que mientras las analógicas puede encontrarse en un instante determinado en cualquiera de un número infinito de estados, las digitales sólo son capaces de alcanzar un número finito de situaciones.

- Por ejemplo, una fuente de información analógica es el nivel de abertura de una puerta. Un ejemplo de fuente de información digital es el resultado de un partido de futbol (1, X o 2).

A.3. Señales digitales y analógicas

- Las señales representan el estado de una fuente de información mediante alguna medida física, medible. Por tanto, las señales son datos y transportan información.
- Si usamos la diferencia de potencial entre dos conductores eléctricos como señal, el grado de abertura de una puerta puede representarse de forma continua si asignamos, por ejemplo, 0 voltios a una puerta cerrada y 5 voltios a una puerta abierta, y el resto de valores intermedios indicando el grado de abertura. En el caso de representar el resultado "quinielístico" de un partido de futbol sólo tres posibles tensiones o rangos de tensiones tendrían significado.

A.4. La ventaja de trabajar en digital

- Todos los medios de transmisión distorsionan las señales que transmiten. Esto provoca que el valor de la señal transmitida difiera en una cierta cantidad de su valor original (el que tenía antes de iniciar la propagación).
- En el caso de las señales analógicas, esta fuente de distorsión se traduce en una fuente de error, o lo que es lo mismo, en una pérdida irreversible de información ya que el receptor no puede saber cuál era el valor original de la señal (en general, cada valor posible de la señal es tan válido como cualquier otro).
- En el caso de las señales digitales, la aparición de ruido durante la transmisión no implica necesariamente una pérdida de información. Si el error no es suficiente como para hacer "saltar" a la señal desde un rango válido hasta otro, el receptor no se equivocará a la hora de interpretar la señal.

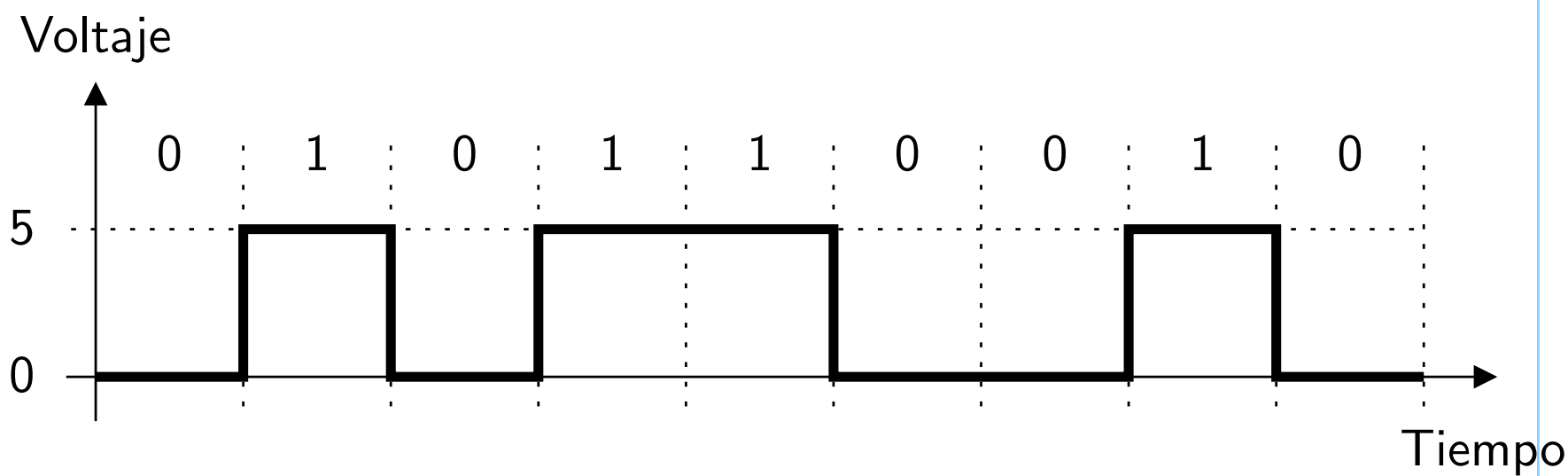
A.5. Amplificadores y repetidores

- Para mejorar la relación entre la energía de la señal de datos y la energía de la señál de ruido puede utililizarse un amplificador de la señal de datos. En la práctica dicho amplificador siempre incrementa ligeramente la energía de la señal de ruido por lo que el número de veces que puede aplicarse este proceso es finito.

- En el caso de una señal digital, la señal puede interpretarse y regenerarse (librarse totalmente del ruido) tantas veces como queramos. Este proceso es el que realiza un repetidor digital.

A.6. Señales binarias y bits

- Por sencillez, la mayoría de los sistemas de tratamiendo de información digital existenten manejan sólo dos posibles estados. Un bit (Binary digIP) es un dato digital y representa generalmente estos estados mediante los símbolos “0” y “1”.
- Cuando una secuencia de bits se representa mediante una señal obtenemos una señal binaria. Por ejemplo, en la figura:



se muestra una señal de tensión binaria que representa a la secuencia de bits 010110010. En ella hemos dispuesto que el bit de información 0 se codifica mediante un potencial de 0 voltios y el bit 1 mediante 5 voltios.

A.7. Conversión analógica/digital

- Una señal analógica puede convertirse en una señal digital y viceversa. En en el primer caso el proceso de digitalización implica en general una pérdida irreversible de información. En el segundo caso sólo existe un cambio de representación.

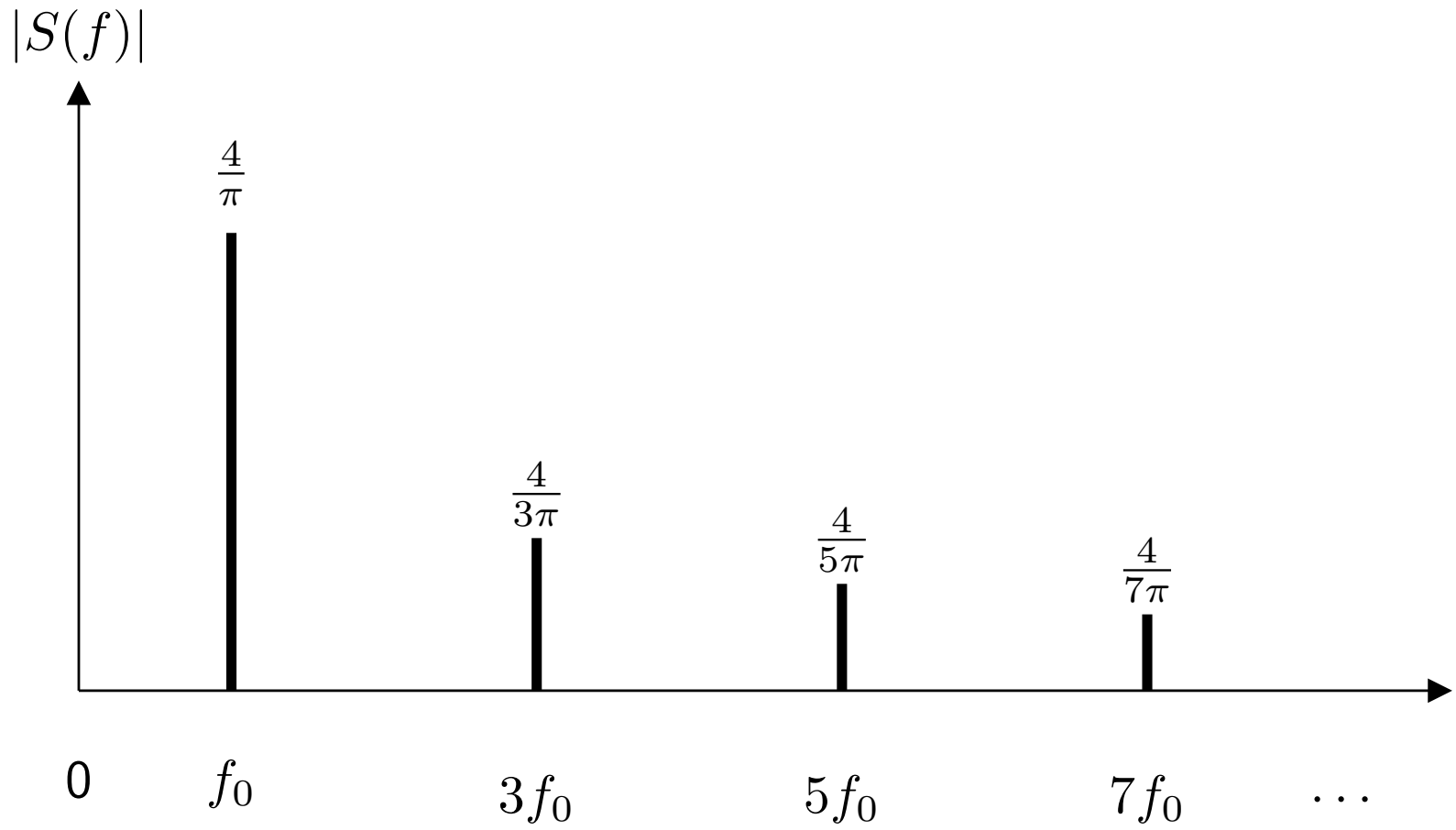
- Una muestra (el valor de la señal en un instante de tiempo) de una señal analógica representa el estado de la fuente de información analógica que dió lugar a esta señal. Puesto que una fuente de este tipo puede encontrarse en un número infinito de estados, una muestra analógica puede también tener una infinidad de valores. En estas circunstancias necesitaríamos un número infinito de bits de datos para representar el valor exácto de la muestra.

- En la práctica esto es imposible (por requerimientos de memoria) y además, no tiene sentido porque si no podemos manipular una señal analógica sin deteriorarla, de nada nos vale dar un significado diferente a cada nivel de señal diferente. Desde el punto de vista de una fuente de información, lo que estamos haciendo es contemplar dicha fuente

suponiendo que sólo puede alcanzar un número finito de estados. A este proceso de le llama **cuantificación**.

A.8. Espectro de una señal digital

- El precio que se paga por usar señales cuantificadas (digitales) es que la tasa de transferencia de información por intervalo de tiempo finito es una tasa finita. Esto significa que la señal digital debe conservar su valor durante un cierto intervalo de tiempo mínimo que permita al receptor reconocer el bit de información transportado.
- Desde el punto de vista del espectro de frecuencias de una señal digital, que en general tiene este aspecto:



la reducción de dicho intervalo temporal implica una expansión en el dominio de la frecuencia de las diferentes componetes de frecuencia.

- Por tanto, a mayor tasa de bits, más ancho de banda es necesario para tranportar la señal digital.

Apéndice B

Perturbaciones en la transmisión de señales digitales

B.1. Efectos de la limitación del ancho de banda

- Las señales gastan energía en su propagación (se atenúan).
- La atenuación depende de la frecuencia porque los medios de transmisión transmiten con menor atenuación un determinado rango de frecuencias. Típicamente se comportan como un filtro paso bajo.
- Para ver de qué manera afecta esto a la transmisión de señales digitales es necesario conocer cómo es el **espectro de frecuencias** o **función de densidad espectral** [16] de una señal digital. Para encontrarlo podemos hacer uso de una herramienta matemática denominada **análisis de Fourier**, que permite representar cualquier señal periódica mediante una sumatoria (a priori infinita) de señales senoidales [19].
- Es evidente que las señales digitales generalmente no son periódicas. Sin embargo, y por suerte para nuestro análisis inicial, la señal digital que mayores requerimientos de ancho de banda posee es aquella que representa a la secuencia de bits $\cdots 010101 \cdots$, que sí genera una

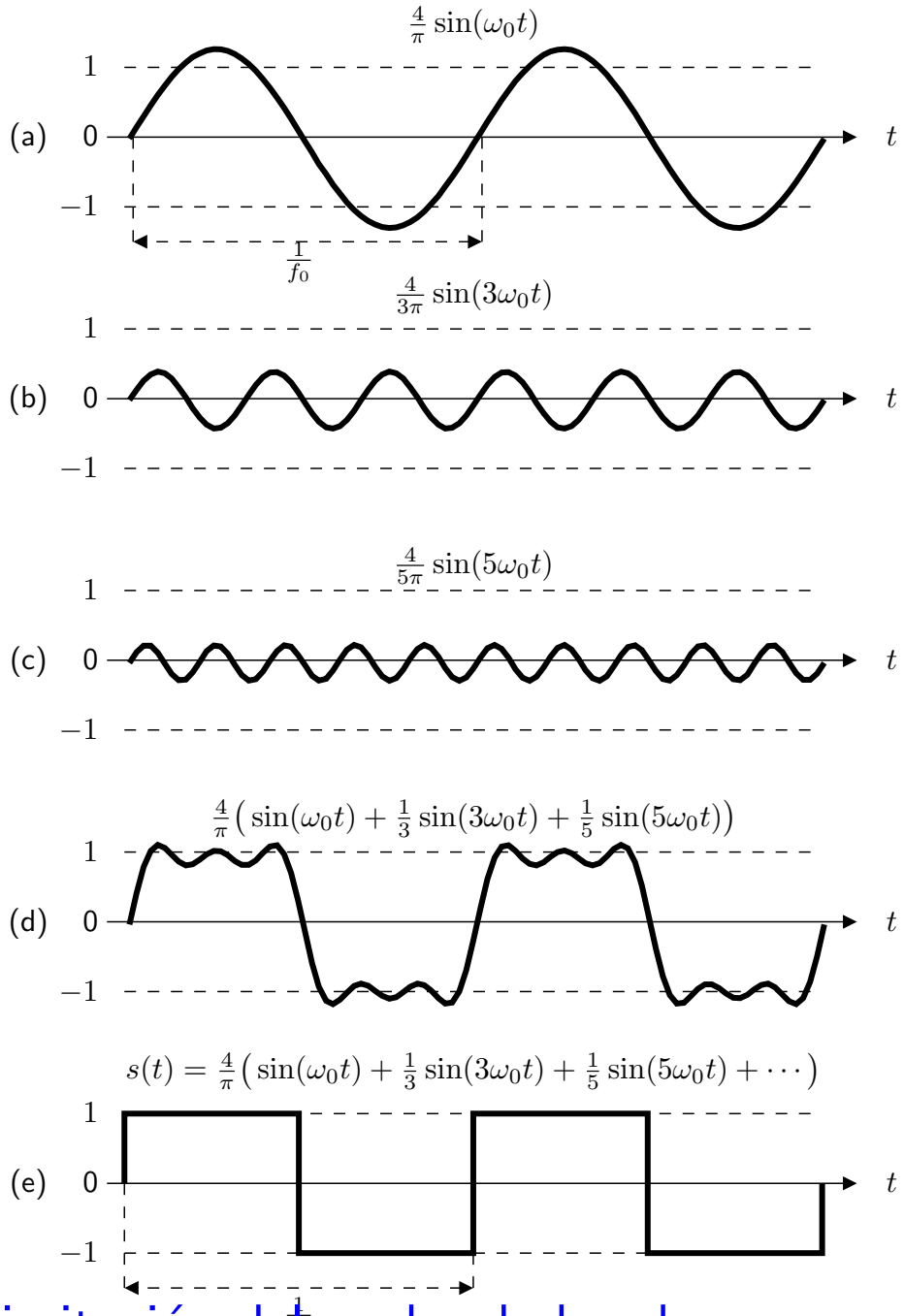
señal periódica. A esta secuencia se la conoce por ello como **la peor secuencia posible** [10], porque genera la señal digital con periodo más corto y por tanto con la componente de frecuencia fundamental ω_0 más alta. Nótese que sólo si la función es periódica se puede hablar de frecuencia fundamental.

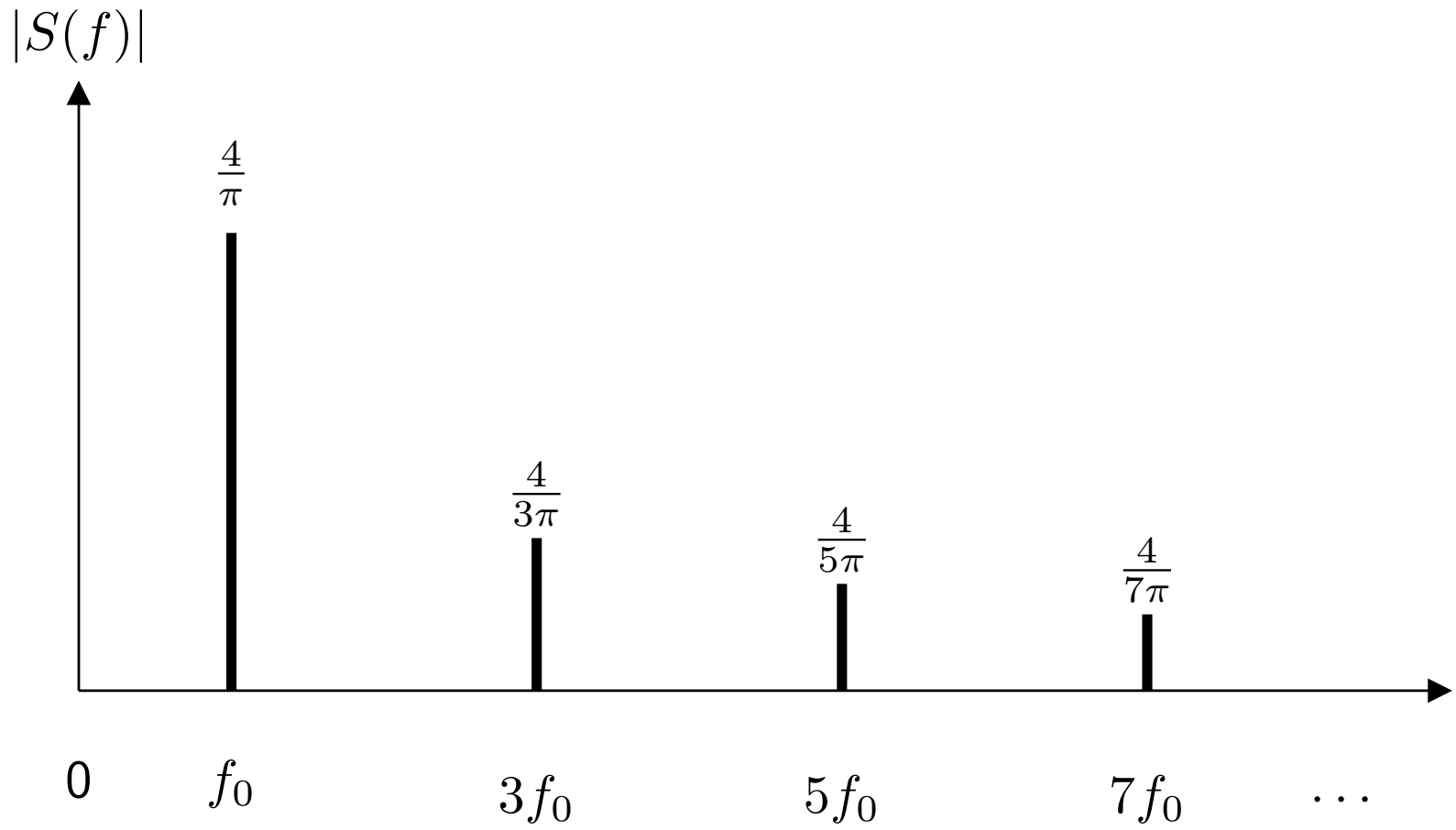
- Supongamos que nuestro sistema de representación utiliza una amplitud de señal 1 para codificar un 1 y -1 para un 0. Bajo estas circunstancias, la figura:

  muestra gráficamente cómo sería el proceso de aproximación mediante la serie de Fourier para la señal digital que representa a la peor secuencia posible, que analíticamente consiste en

$$s(t) = \frac{4}{\pi}\left(\sin(\omega_0 t) + \frac{1}{3}\sin(3\omega_0 t) + \frac{1}{5}\sin(5\omega_0 t) + \cdots\right). \qquad \text{(B.1)}$$

- Como puede apreciarse, todas las componentes de frecuencia que son múltiplos pares de ω_0 (la frecuencia fundamental) son nulas, lo que genera un espectro de frecuencias disperso (no denso) característico de todas las señales periódicas. Gráficamente:


$\frac{4}{\pi}\sin(\omega_0 t)$
1
(a) 0
-1
t
$\frac{1}{f_0}$
$\frac{4}{3\pi}\sin(3\omega_0 t)$
1
(b) 0
-1
t
$\frac{4}{5\pi}\sin(5\omega_0 t)$
1
(c) 0
-1
t
$\frac{4}{\pi}\big(\sin(\omega_0 t) + \frac{1}{3}\sin(3\omega_0 t) + \frac{1}{5}\sin(5\omega_0 t)\big)$
1
(d) 0
-1
t
$s(t) = \frac{4}{\pi}\big(\sin(\omega_0 t) + \frac{1}{3}\sin(3\omega_0 t) + \frac{1}{5}\sin(5\omega_0 t) + \cdots\big)$
1
(e) 0
-1
t
$\frac{1}{f_0}$



donde $\omega_n = 2\pi f_n$ es la n-ésima componente de frecuencia de $s(t)$ expresada en radianes por segundo (f_n se mide por tanto en Hercios o Hz)[1] y donde $S(f)$ denota a la transformada de Fourier de $s(t)$ [16,

[1] En honor al físico alemán Heinrich Rudolf Hertz.

19]. En nuestro ejemplo ocurre que todos los a_n son cero, incluyendo a_0 que coincide con la media de la señal.

- Este espectro posee dos propiedades importantes de cara a su transmisión:

  1. El ancho de banda que consume una señal digital es infinito, es decir, que para que el receptor reciba una señal ideal es necesario que el ancho de banda del enlace sea ilimitado (no filtre ninguna frecuencia).
  2. La mayor parte de la energía se concentra en las bajas frecuencias, especialmente en la componente f_0. De hecho, si sólo se transmite la componente de frecuencia fundamental todavía es posible la recepción con éxito. Por ejemplo, en la figura que muestra la aproximación de Fourier a la peor secuencia posible se aprecia que cuando la frecuencia fundamental es f_0 Hz se pueden transmitir hasta $2f_0$ bps.

B.1.1. Estimación de Nyquist para la capacidad de un enlace

- En ausencia de ruido, Nyquist estimó que es posible transmitir una señal digital usando únicamente su componente de frecuencia fundamental.
- Si w es el ancho de banda del medio y m es el número de niveles de señalización, la capacidad c_{Ny} máxima del medio (la máxima tasa de bits a la que es posible transmitir) medida en bits/segundo o bps es igual a:

$$c_{\mathrm{Ny}} = 2w \log_2(M) \tag{B.2}$$

Ejemplo B.12: El ancho de banda de un enlace telefónico posee típicamente unos 3.000 Hz, dispuestos desde los 300 hasta los 3.300 Hz. Suponiendo que se están utilizando 2 niveles de señalización, estimar la máxima velocidad de transferencia que es posible alcanzar (capacidad de Nyquist).

$$C_{\mathrm{Ny}} = 2 \times 3.000 \times \log_2(2) = 6.000 \text{ bps.}$$

B.2. Efectos del ruido y de la atenuación

- La atenuación de la amplitud de las señales es directamente proporcional a la distancia recorrida [8].[2]
- Sin embargo, el ruido es inherente al medio y por tanto no es atenuado. Esto provoca que en algún punto de la distancia recorrida la relación entre la potencia de la señal y la potencia del ruido no sea suficiente como para distinguir la señal de datos de la señal de ruido.
- Esta situación puede solventarse utilizando repetidores.

B.2.1. Estimación de Shannon-Hartley para la capacidad de un enlace

- Debido a ruido y a la atenuación, la tasa máxima de transmisión está limitada.

[2] Nótese que en términos de potencia la atenuación es exponencial (recuérdese que $V^2 = PR$ donde V denota nivel de tensión, P potencia y R resistencia).

- Shannon y Hartley estimaron que en presencia de ruido blanco gausiano [16], la capacidad c_{SH} medida en bps de un medio con w Hz de ancho de banda depende de la relación señal/ruido o SNR (Signal-to-Noise Ratio) según la expresión:

$$c_{\mathrm{SH}} = w \log_2(1 + \mathrm{SNR}) \tag{B.3}$$

Ejemplo B.13: Calcúlese la capacidad de un enlace telefónico ($W = 3.000$ Hz) que posee una SNR de 35 dB.

$$\mathrm{SNR[dB]} = 10 \times \log_{10}(\mathrm{SNR}),$$

y por tanto,

$$\mathrm{SNR} = 3.162.$$

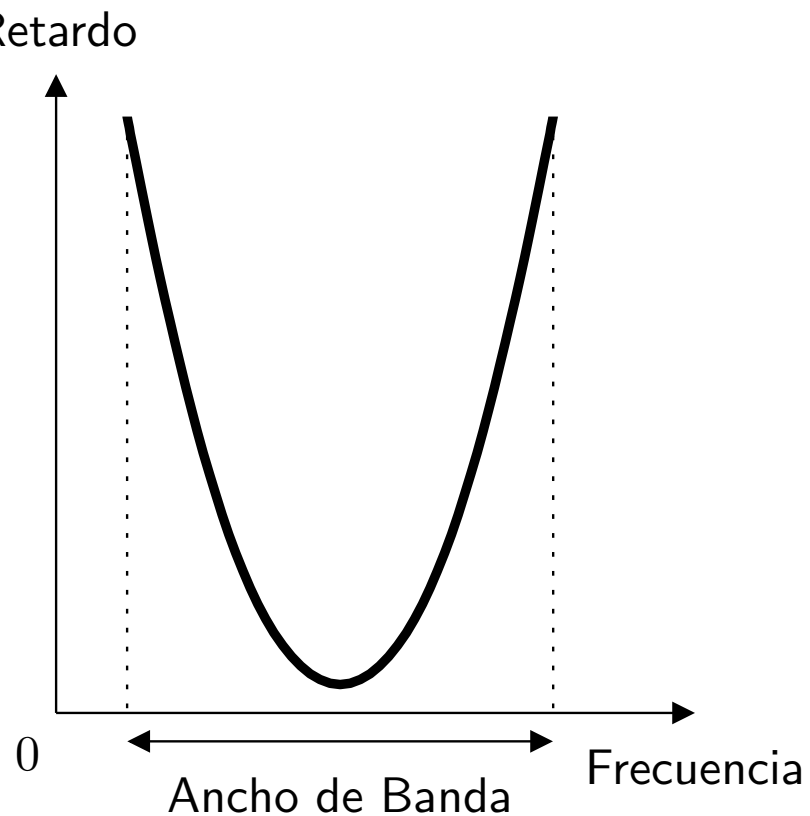
Así, la capacidad del canal sería

$$C_{\mathrm{SH}} = 3.000 \times \log_2(1 + 3.162) = 34.860 \text{ bps}.$$

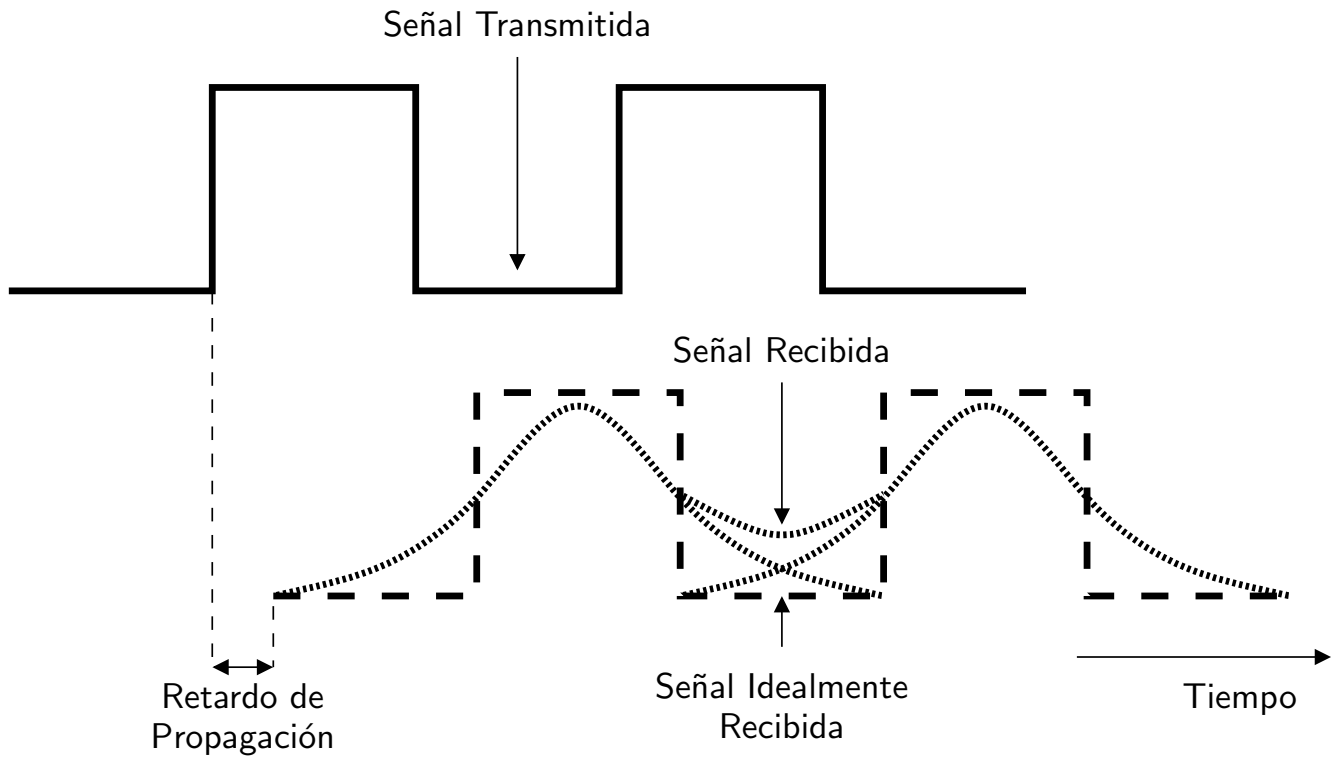
Si se quiere transmitir más rápido es necesario incrementar el ancho de banda o disminuir el nivel de ruido.

B.3. Efectos de la distorsión de retardo

- La velocidad de propagación de las señales electromagnéticas depende de su frecuencia [27]. Típicamente el medio transmite con mayor velocidad aquellas frecuencias que están próximas al intervalo de frecuencias que el medio menos atenúa:



- Si los elementos de señalización están suficientemente próximos en el tiempo, este efecto puede provocar que las componentes de frecuencia más bajas y altas de un elemento se mezclen en el receptor con las componentes de frecuencia intermedias del siguiente elemento:



- La distorsión de retardo (también llamada interferencia entre símbolos) es directamente proporcional a la velocidad de transmisión y a la

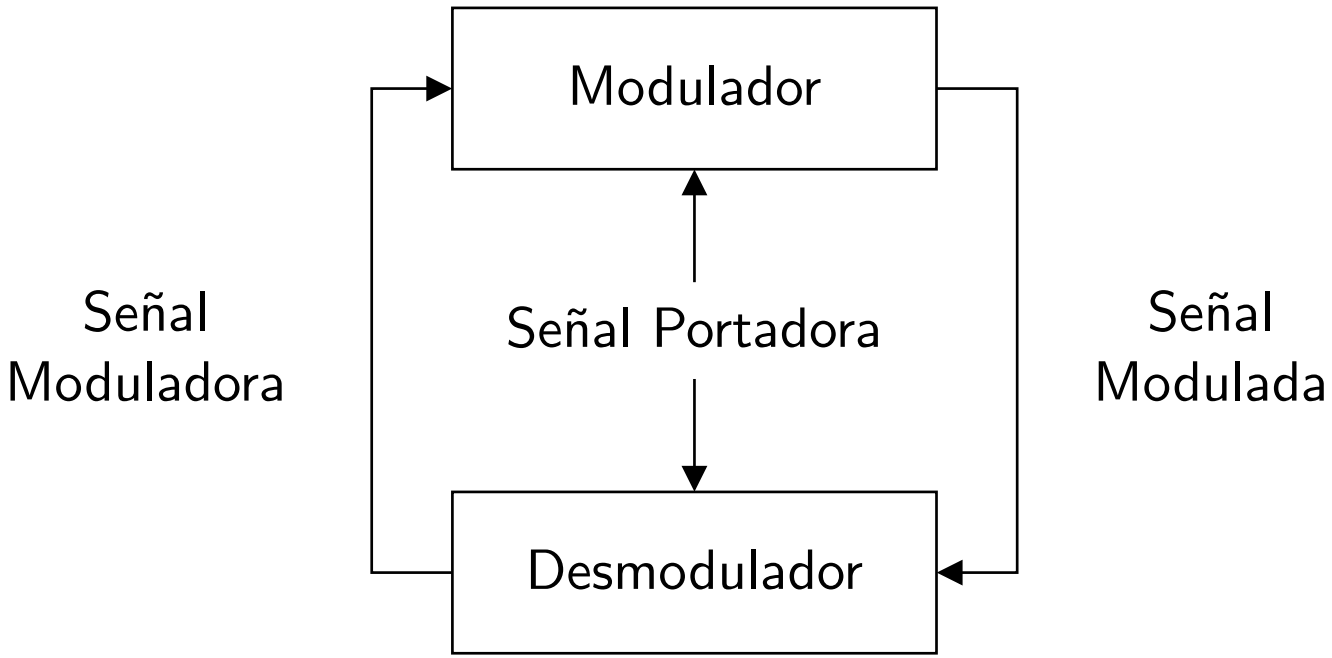
distancia recorrida [10].

Apéndice C

Modulación de señales digitales

C.1. La modulación de señales

- La modulación es el proceso mediante el cual una señal (que no tiene por qué ser digital) transfiere la información que transporta a otra, que inicialmente no aportaba información [19, 16].

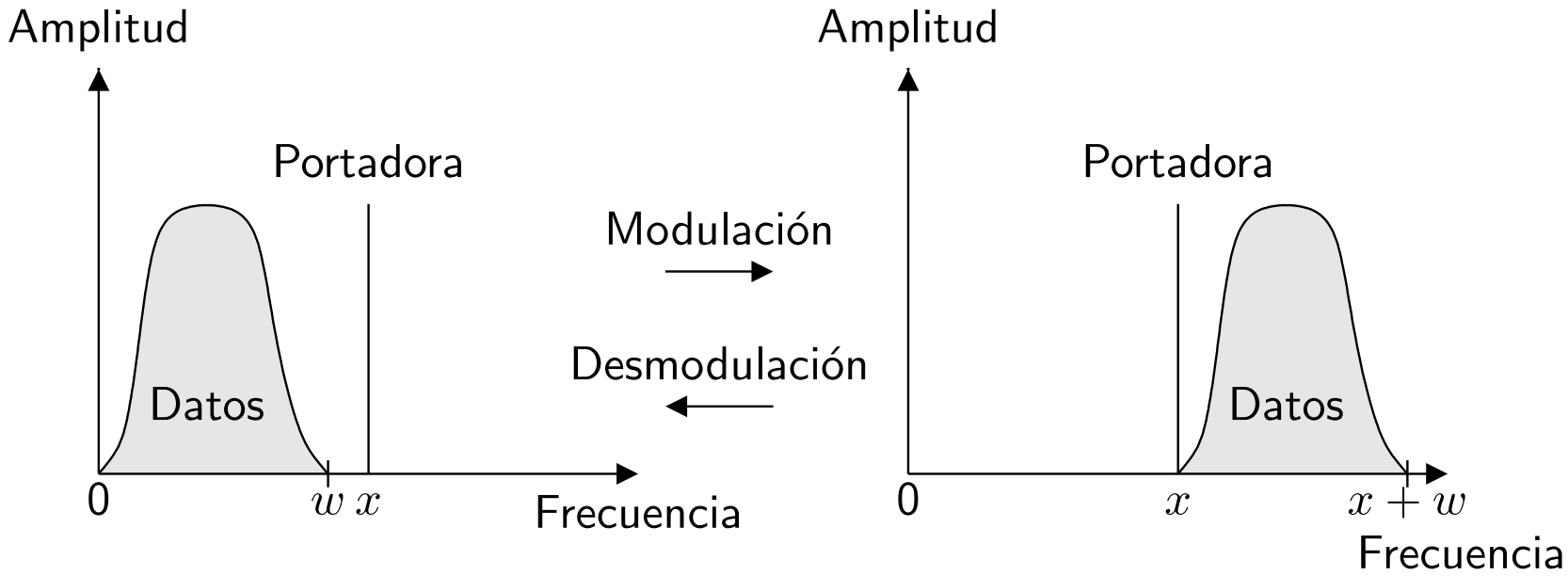


- En la modulación intervienen tres señales diferentes:

    1. **La señal moduladora** (o señal de datos) que contiene la información que se desea transmitir.
    2. **La señal portadora** que debe poseer una amplitud, frecuencia y fase constantes o cambiantes de forma predecible. Típicamente es una señal senoidal.
    3. **La señal modulada** que es la señal resultante de modificar la señal portadora en función de la señal moduladora. La señal modulada contiene la misma información que la señal moduladora.

C.2. Utilidad de la modulación de señales

1. **Permite el diseño de sistemas de multiplexación en frecuencia.** La modulación permite desplazar el rango de frecuencias de la señal a transmitir al canal designado.



2. **Hace posible la radiación efectiva de señales.** Sólo se upede radiar al espacio una señal de forma efectiva si la antena radiadora es del orden de un décimo o más de la longitud de onda más larga en la señal [16].

C.3. Bits de datos, elementos de señalización y Baudios

- **Bit (Binary digIT) de dato:** es el símbolo asociado a uno de los estados de una fuente de información binaria.
- **Elemento de señalización:** es el trozo de señal que representa a un símbolo (binario o n-ario).
- **Baudio:** es la medida de la tasa de señalización.

Ejemplo C.14: Antiguamente se hablaba de que un modem podía transmitir a 1.200 Baudios, lo que significa que se transmitían 1.200 elementos de señalización en 1 s. Nótese que esto no implica necesariamente que se transmitieran 1.200 bps. Esto en realidad dependía del número de elementos de señalización diferentes.

C.4. Modulación binaria en amplitud o ASK

- En ASK (Amplitude-Shift Keying) la señal de datos controla la amplitud de la señal portadora.

C.4.1. Modulación

- Si $s(t)$ representa a una señal de datos binaria unipolar (que nunca se hace negativa) en **banda base**[1] (que su espectro de frecuencias ocupa el rango $[0, w_s]$ donde w_s es su máxima componente de frecuencia angular) y $c(t)$ es la portadora, la señal modulada $s_{\mathrm{ASK}}(t)$ puede generarse mediante el producto de señales

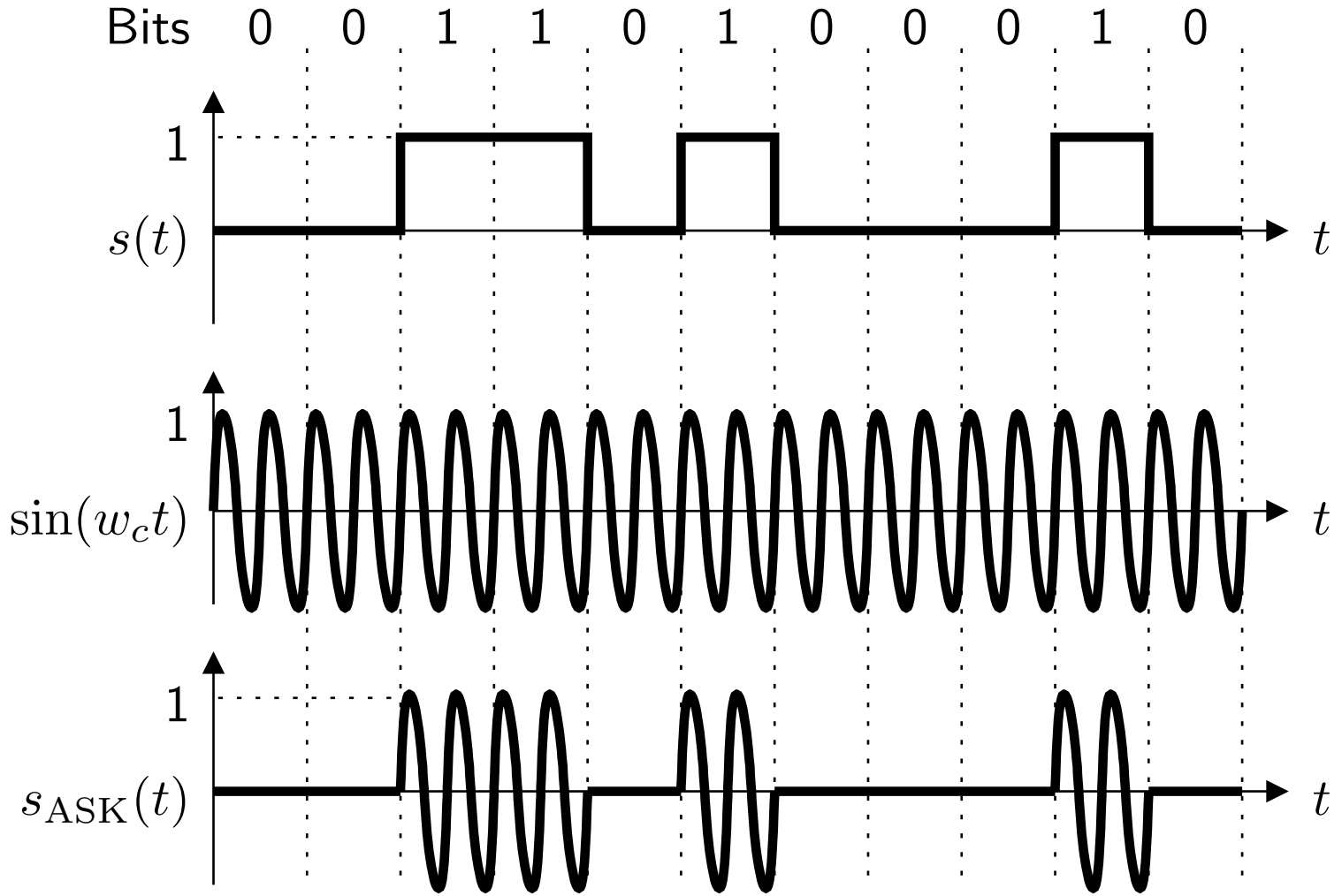
$$s_{\mathrm{ASK}}(t) = s(t)c(t). \tag{C.1}$$

[1] Una señal digital binaria unipolar en banda base puede ser descrita también en el dominio del tiempo como una modulación por pulsos donde típicamente el bit 1 se representa con una amplitud distinta de 0 y constante, y el bit 0 mediante una amplitud 0 [16].

- Como ya hemos indicado, la señal $c(t)$ es normalmente una senóide de la forma

$$c(t) = A_c \sin(w_c t + \phi) \tag{C.2}$$

donde por comodidad de exposición supondremos que $A_c = 1$ (para simplificar las expresiones y obtener espectros normalizados) y $\phi = 0$ (porque en ASK la fase es irrelevante para el receptor). En la siguiente figura se muestran las tres señales:



- Usando la expresión en serie de Fourier, , la señal digital que representa a la peor secuencia posible $s(t)$ puede ser expresada como

$$s(t) = 1 + \frac{4}{\pi} \sum_{n=0}^{\infty} \frac{\sin\left((2n+1)w_o t\right)}{2n+1}$$

y por tanto, tras realizar el producto obtenemos

$$s_{\mathrm{ASK}}(t) = \sin(w_c t) + \frac{4}{\pi} \sum_{n=0}^{\infty} \frac{\sin\left((2n+1)w_0 t\right)}{2n+1} \sin(w_c t).$$

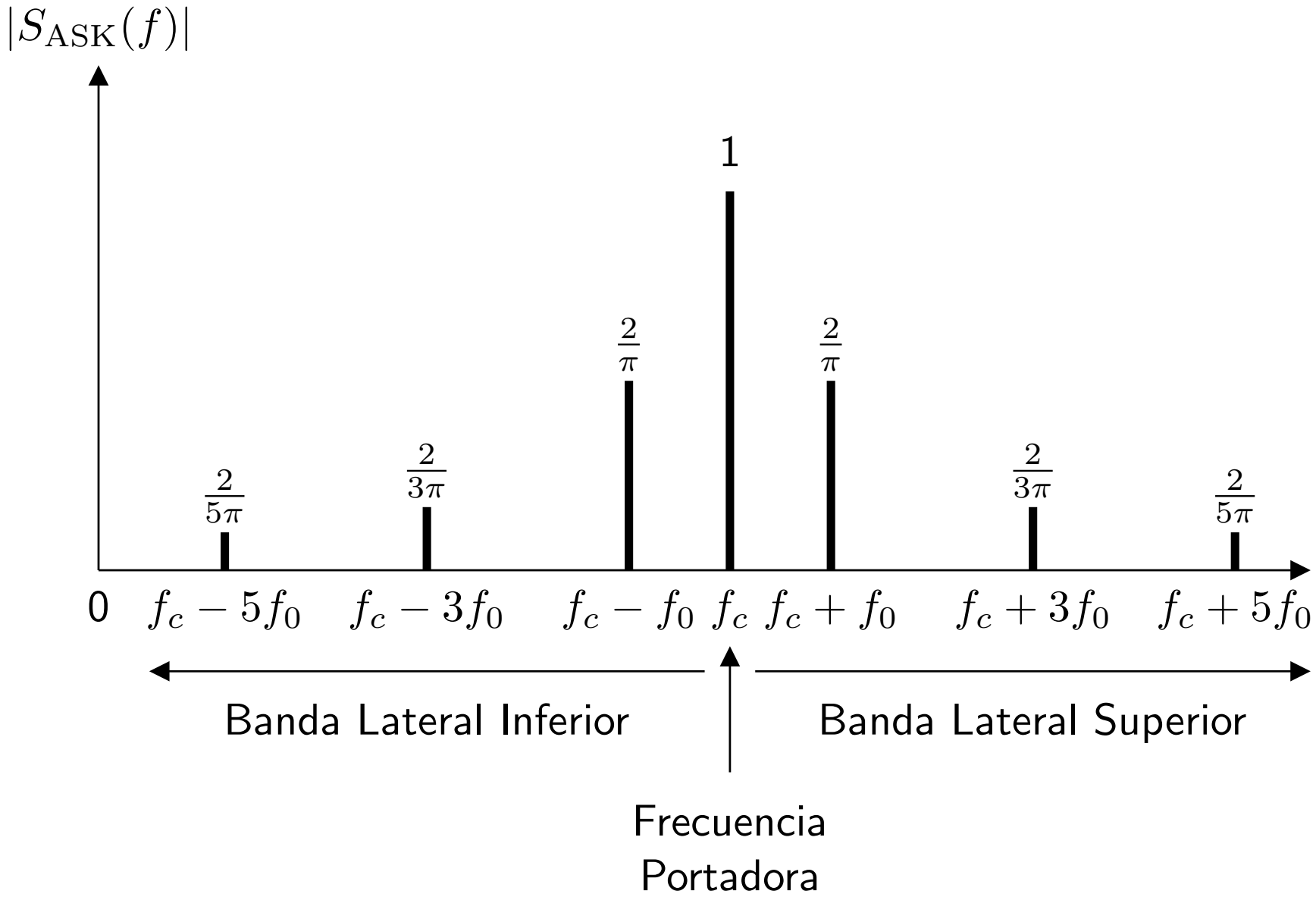
Si ahora aplicamos el cambio trigonométrico

$$2\sin(\alpha)\sin(\beta) = \cos(\alpha-\beta) - \cos(\alpha+\beta) \qquad \text{(C.3)}$$

obtenemos que

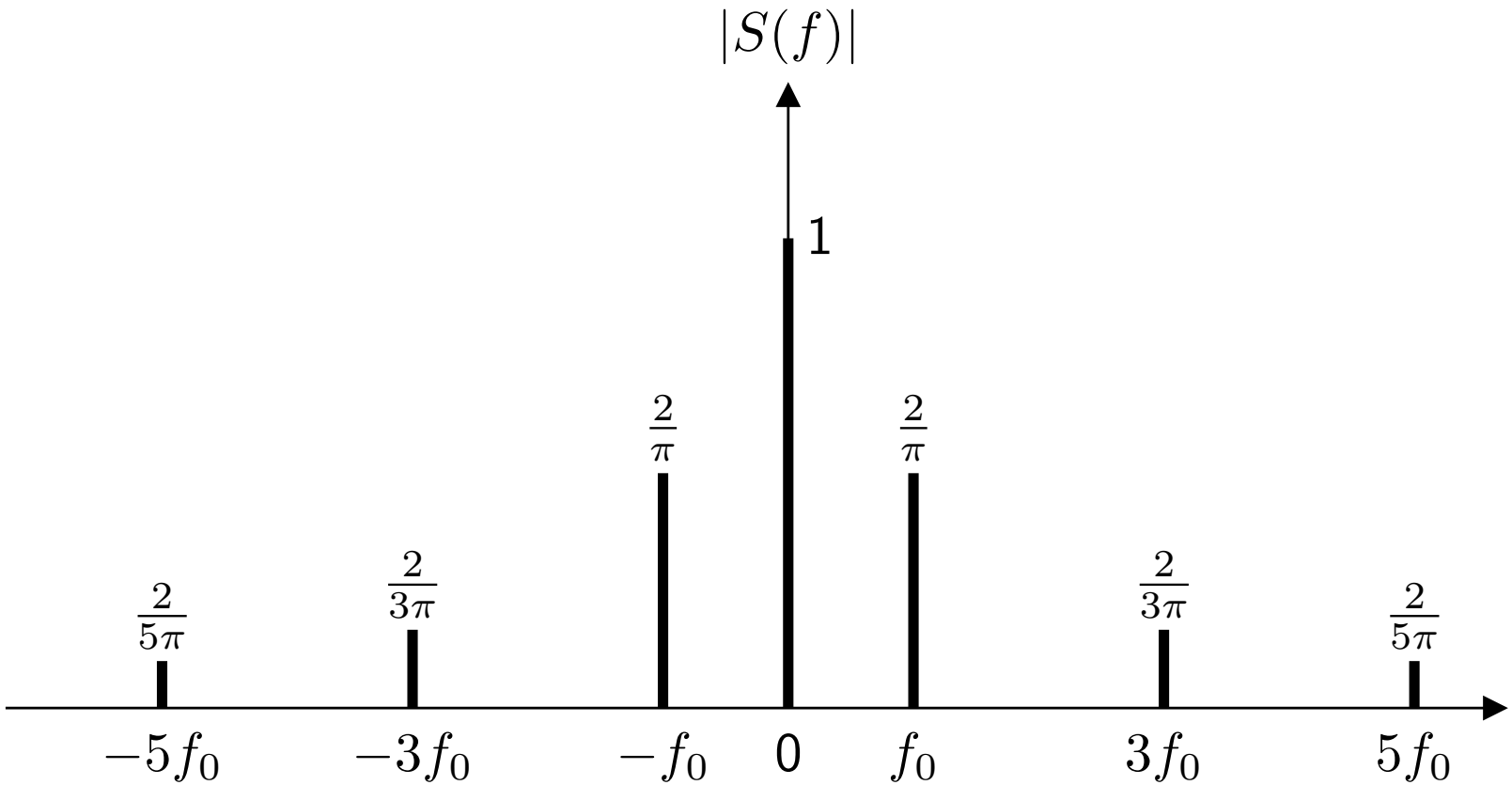
$$s_{\mathrm{ASK}}(t) = \sin(w_c t) + \frac{4}{2\pi} \sum_{n=0}^{\infty} \frac{\cos\left((2n+1)w_0 t - w_c t\right) - \cos\left((2n+1)w_0 t + w_c t\right)}{2n+1}$$

de donde se deduce que el espectro $|S_{\mathrm{ASK}}(f)|$ es el que se muestra en la Figura:

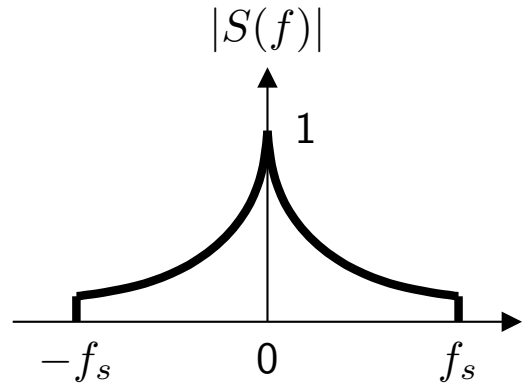

$|S_{\text{ASK}}(f)|$
1
$\frac{2}{\pi}$
$\frac{2}{\pi}$
$\frac{2}{5\pi}$
$\frac{2}{3\pi}$
$\frac{2}{3\pi}$
$\frac{2}{5\pi}$
0
$f_c - 5f_0$
$f_c - 3f_0$
$f_c - f_0$
f_c
$f_c + f_0$
$f_c + 3f_0$
$f_c + 5f_0$
Banda Lateral Inferior
Banda Lateral Superior
Frecuencia
Portadora

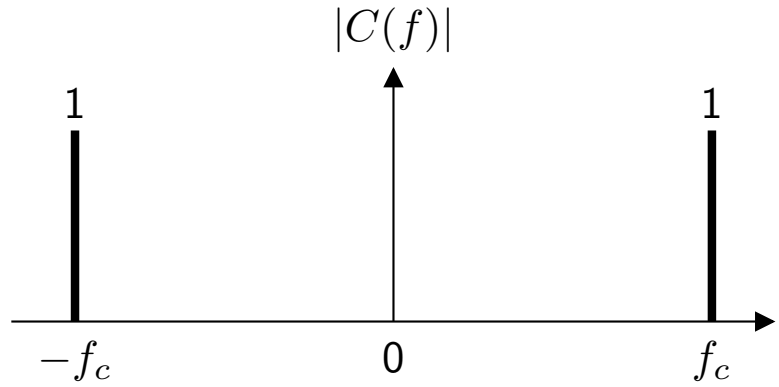
- Si comparamos $|S_{\mathrm{ASK}}(f)|$ con $|S(f)|$ comprobaremos que el ancho de banda se ha duplicado y que el espectro de $s(t)$ ahora aparece desplazado f_c Hz, aunque atenuado en un factor de 2. Nótese además que ambas bandas laterales transportan la misma información porque son simétricas respecto de f_c (la frecuencia de la portadora). Esto significa que podemos transmitir solo una de ellas y la desmodulación todavía sería posible. Para este propósito se suele utilizar un filtro paso-banda con frecuencias de corte $]f_c, f_c + f_s]$ (eliminando por tanto la banda lateral inferior) y a la señal filtrada se le conoce por señal modulada ASK en banda lateral única, en adelante $s_{\mathrm{ASK\_BLU}}(t)$. Como el filtro elimina la mitad del espectro (que es simétrico) también elimina la mitad de la energía. Por tanto, el filtro suele tener una ganancia de 2 para que al final la SNR obtenida al transportar $s_{\mathrm{ASK}}(t)$ o $s_{\mathrm{ASK\_BLU}}(t)$ sea la misma.

- Para recuperar $s(t)$ a partir de $s_{\mathrm{ASK}}(t)$ o de $s_{\mathrm{ASK\_BLU}}(t)$ (desmodular) basta con multiplicar de nuevo la señal modulada por la portadora. Para demostrar esto ahora vamos a utilizar los espectros extendidos de las señales en lugar de su representación en el dominio del tiempo.

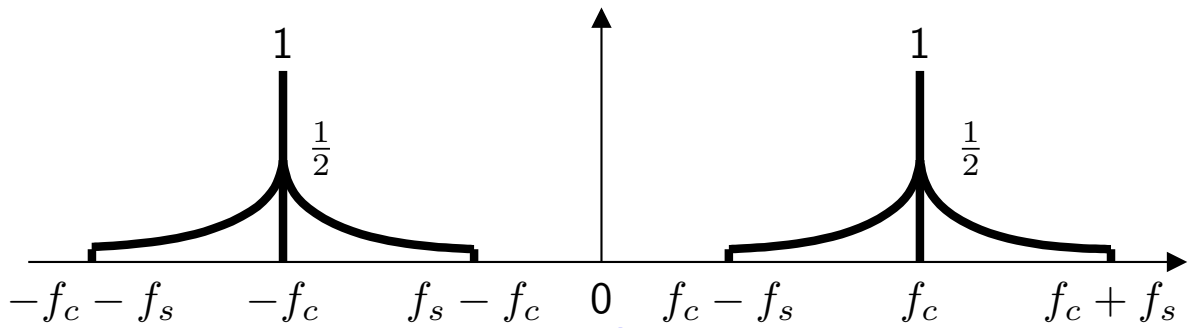


- Teniendo en cuenta dicha representación, la multiplicación en el dominio del tiempo de $s(t)$ y $c(t)$ equivale a desplazar el espectro de $s(t)$ hasta centrarlo en la frecuencia de la portadora.
- En la siguiente figura se resume todo el proceso de modulación ASK (que coincide con la modulación en amplitud de señales analógicas o AM – Amplitude Modulation – [16]).



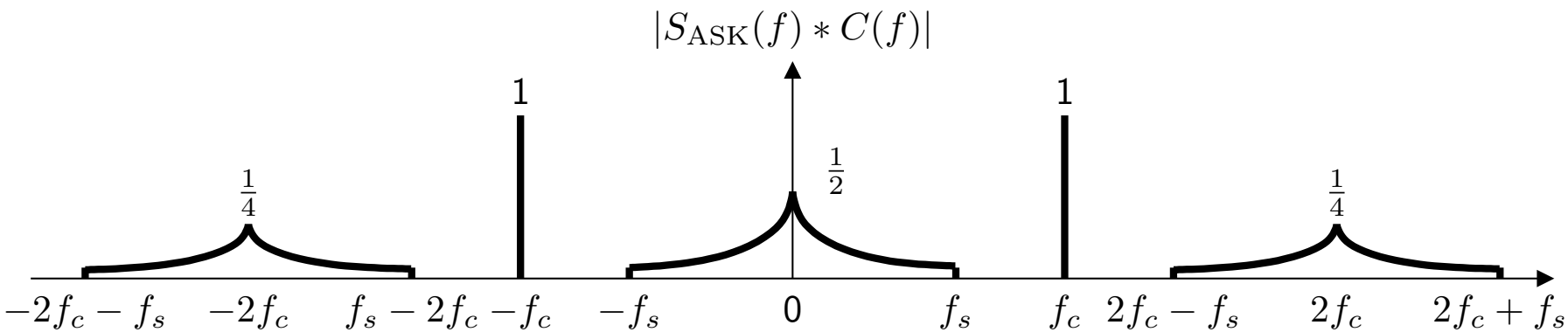


$$|S_{\mathrm{ASK}}(f)| = |S(f) * C(f)|$$



C.4.2. Desmodulación

- Si volvemos a multiplicar la señal modulada de nuevo por la portadora, crearemos dos réplicas desplazadas sobre f_c y $-f_c$ del espectro $|S_{\text{ASK}}(t)|$. Así creamos tres réplicas de $|S(f)|$ en $-2f_c$, 0 y $2f_c$ Hz, pero nótese que la segunda de ellas es la suma de dos espectros idénticos y por lo tanto tiene la misma forma aunque el doble de amplitud que sus lóbulos hermanos. En la siguiente figura se muestra dicho espectro.



- Aplicando a continuación un filtro paso-bajo con frecuencia de corte en f_s y ganancia dos, recuperamos la señal de datos $s(t)$.

C.5. Modulación binaria en frecuencia o FSK

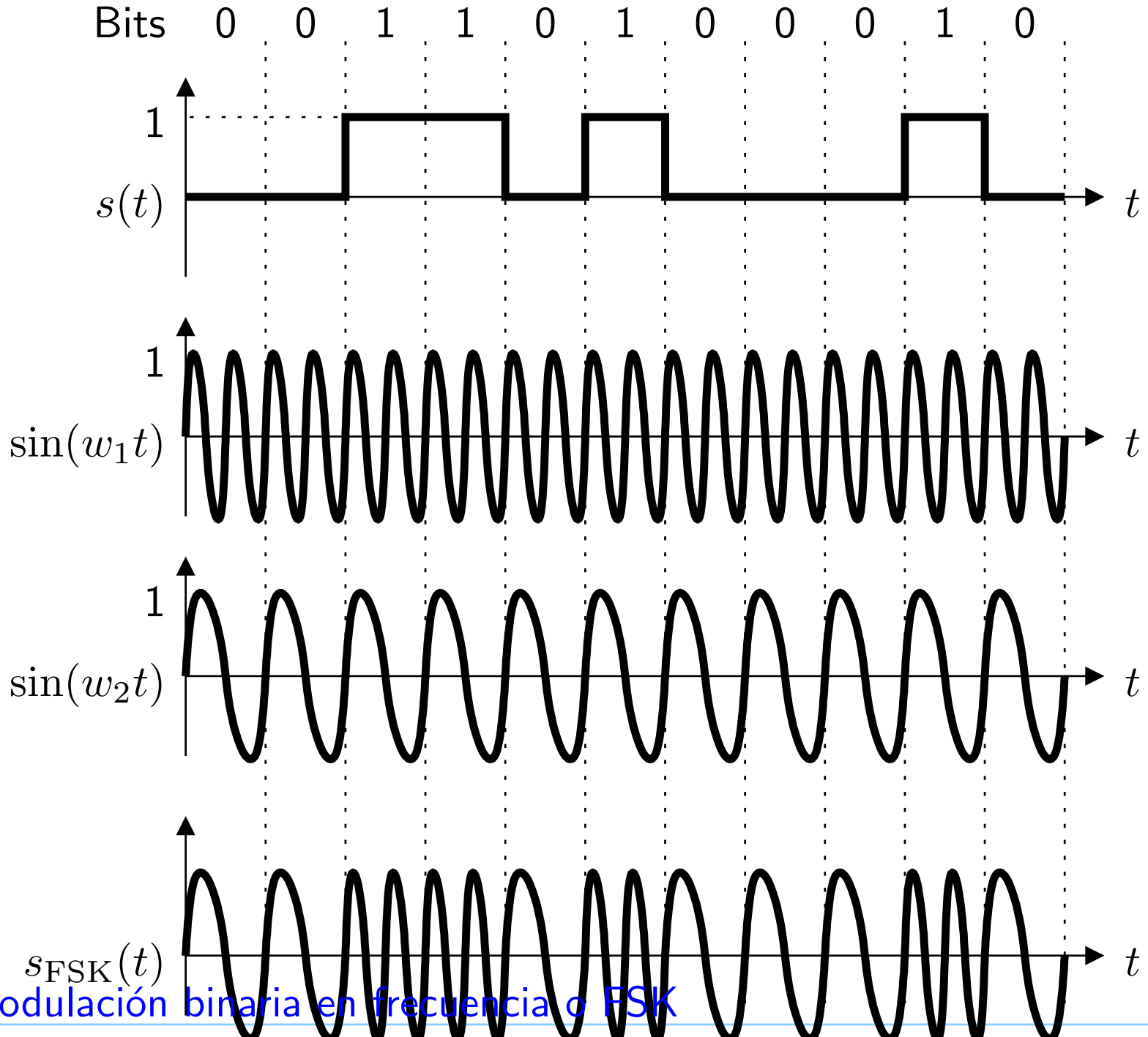
- En FSK (Frecuency-Shift Keying) la señal de datos modifica la frecuencia de la portadora.

C.5.1. Modulación

- Se basa en usar dos portadoras de frecuencias distintas y la misma amplitud. Esto se puede conseguir mediante la relación

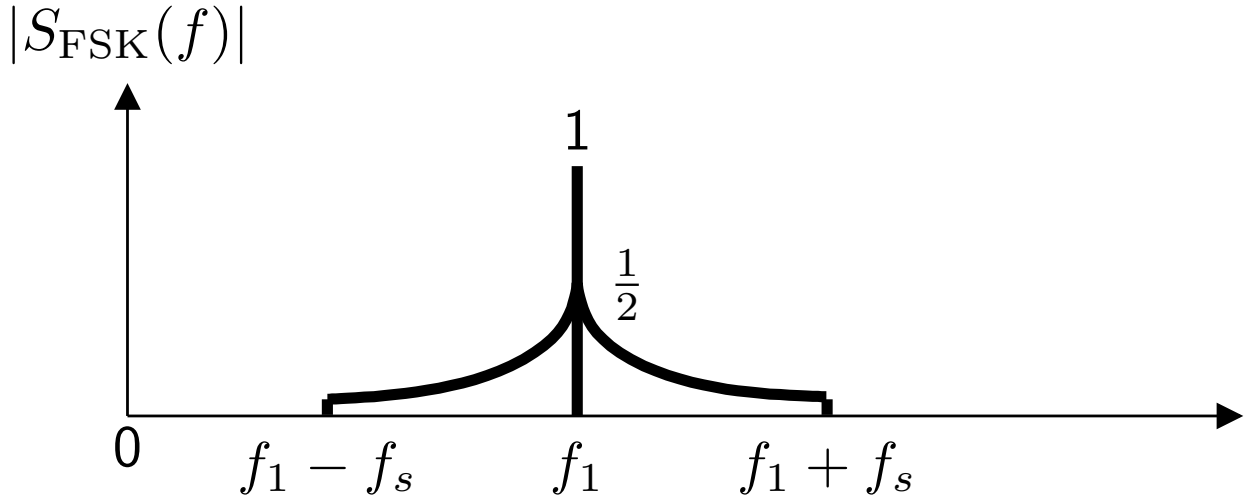
$$s_{\mathrm{FSK}}(t) = s(t)\sin(w_1 t) + \big(1 - s(t)\big)\sin(w_2 t), \qquad \text{(C.4)}$$

siendo $s(t)$ una señal unipolar en banda base con amplitud máxima 1. Ejemplo:

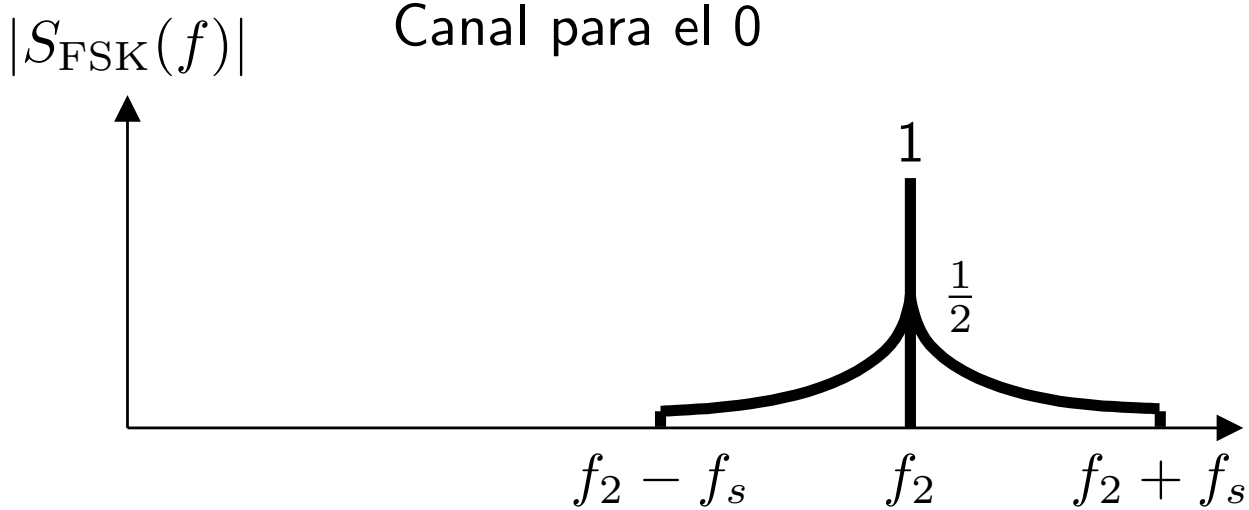

Bits 0 0 1 1 0 1 0 0 0 1 0
1
$s(t)$
t
1
$\sin(w_1 t)$
t
1
$\sin(w_2 t)$
t
$s_{\mathrm{FSK}}(t)$
t

- Como puede apreciarse, la señal FSK generada por la Expresión C.4 es equivalente a sumar la salida de dos moduladores ASK con portadoras de frecuencia w_1 y w_2 [10], pero es importante darse cuenta de que estos moduladores nunca emiten señal al mismo tiempo. Cuando modulamos un 0 es el modulador ASK con portadora w_1 el que emite señal y cuando modulamos un 1 es el modulador ASK con portador w_2.

- Desde el punto de vista de la frecuencia, lo que obtenemos son los espectros mostrados en la figura:



Canal para el 0



Canal para el 1

Cuando modulamos un 0 se utiliza la banda de frecuencias $[f_1 - f_s, f_1 + f_s]$ y cuando modulamos un 1 la banda $[f_2 - f_s, f_2 + f_s]$. Al

igual que ocurre con ASK-BLU, en FSK podemos filtrar la mitad de cada lóbulo para reducir el ancho de banda consumido. Si observamos la figura anterior podemos ver que para el lóbulo que transmite un 0 podemos seleccionar la banda lateral superior y para el lóbulo que transmite un 1 la inferior. Sin embargo, debe verificarse que

$$f_1 + f_s < f_2, \tag{C.5}$$

o de lo contrario el receptor no sabría cuándo se está transmitiendo un 0 o un 1.

- Esto significa que los requerimientos de ancho de banda para FSK son ligeramente superiores a los de ASK, aunque el incremento (normalmente pequeño) en la cantidad de ancho de banda requerido está en la práctica más que justificado porque el ruido del canal tiende a afectar más a la amplitud de la señal modulada que a su frecuencia, lo que provoca que FSK sea más resistente al ruido que ASK.

C.5.2. Desmodulación

- La desmodulación de una señal FSK es la misma que para una ASK, aunque hacen falta dos desmoduladores diferentes, uno para cada portadora.

C.6. Modulación binaria en fase o PSK

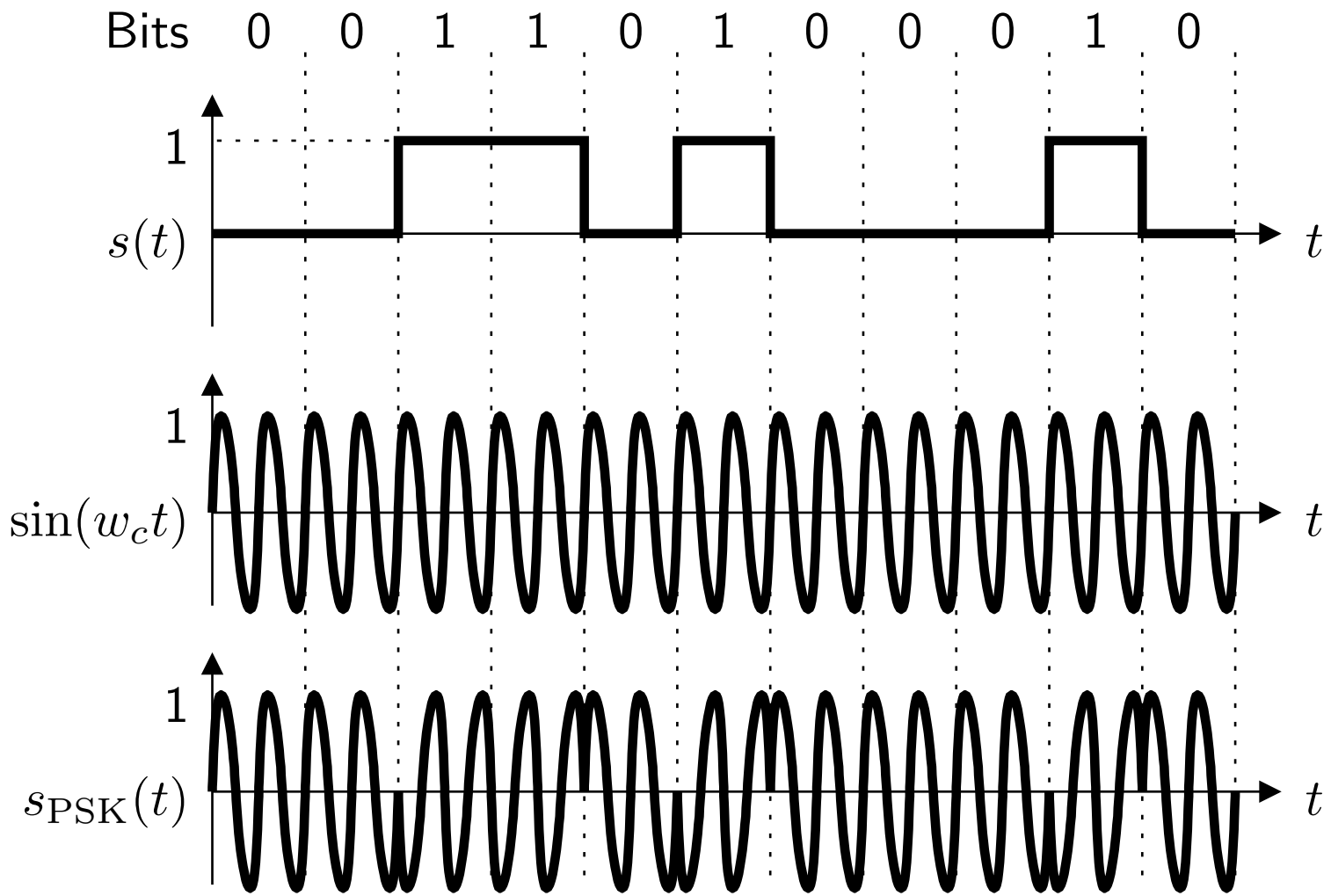
- En PSK (Phase-Shift Keying) la señal de datos modifica la fase de la portadora.

C.6.1. Modulación

- Se puede conseguir si empleamos una modulación ASK pero partiendo de una señal bipolar (en lugar de unipolar). Podemos conseguir fácilmente la versión bipolar $s'(t)$ de una señal de datos a partir de su versión unipolar $s(t)$ utilizando la expresión

$$s'(t) = 2s(t) - 1. \tag{C.6}$$

Ejemplo:



- El espectro de una señal PSK de este tipo es idéntico al de la modulación ASK (véase la figura del espectro de una señal ASK), pero la señal portadora no está presente ya que la media de la señal $s'(t)$ es 0. Los anchos de banda requeridos por ASK y PSK son idénticos.

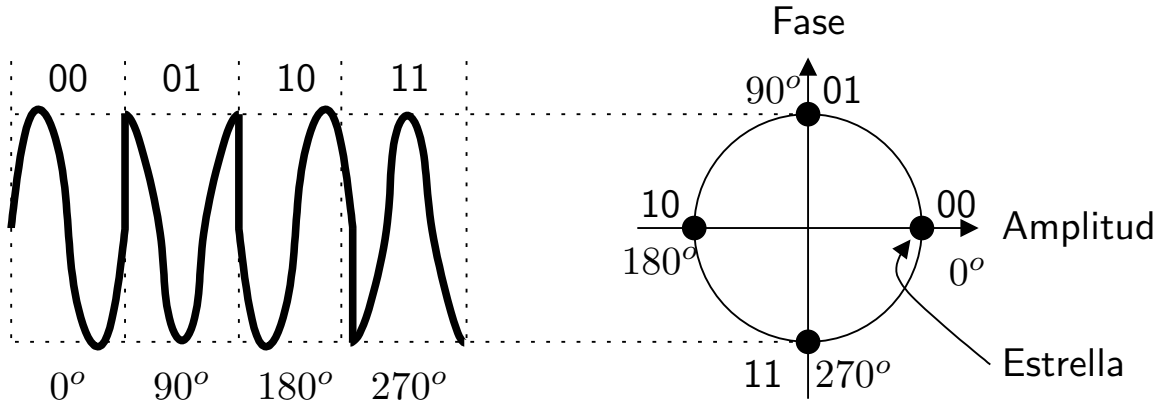
C.6.2. Desmodulación

- Igual que en ASK.

C.7. Modulación n-ária

- En la práctica, los moduladores binarios son demasiado conservadores respecto al nivel de ruido que presenta el canal de transmisión. Esto significa que sería posible usar más de dos niveles de señalización sin que aumentara significativamente la tasa de errores de transmisión. Con el objetivo de aumentar la capacidad del canal, se utilizan moduladores con más de dos niveles de señalización que se basan en:

  1. **Discretizar más finamente los cambios de estado de la portadora**. Por ejemplo, se han diseñado modems (nombre que recibe el dispositivo que MOdula y DEsModula) PSK que funcionan con 4 cambios de fase distintos (0^o, 90^o, 180^o y 270^o), lo que permite transmitir símbolos de 2 bits con cada elemento de señalización:



Constelación 4-PSK

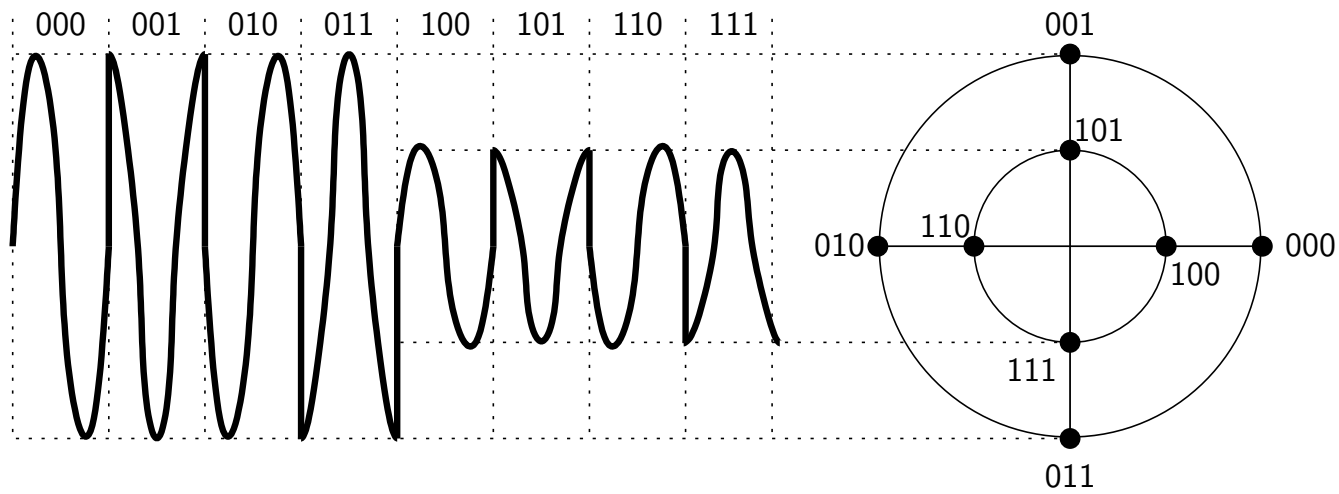
En este caso la tasa de señalización (Baudios) es la mitad que la tasa de bits (bps).

Este tipo de modulación se conoce como **modulación de cambio de fase en cuadratura** [10] o QPSK (Quadrature-Phase-Shift Keying). También se ha llamado 4-PSK.

Existe además otra forma interesante de representar el funcionamiento de un modem "tipo PSK" mediante su diagrama de fase o constelación. Este gráfico enfrenta la amplitud de la portadora frente a la fase. El resultado para 4-PSK se muestra también en la figura anterior.

2. **Modificar simultáneamente más de un parámetro de la**

portadora. Una de las configuraciones que mejor funcionan en la práctica consiste en mezclar las modulaciones PSK y ASK. Dicha combinación se denomina **modulación de amplitud en cuadratura** o QAM (Quadrature Amplitude Modulation). La idea consiste en trabajar con 2^x amplitudes y 2^y fases para representar hasta $(x + y)$ bits por cada elemento de señalización. En la siguiente figura se muestra la modulación 8-QAM, donde $x = 1$ e $y = 2$, siendo en este caso la tasa de bits 3 veces superior a la de Baudios.



Constelación 8-QAM

Cuando se diseña un sistema de modulación QAM (el más uti-

lizado en la actualidad), lo que se intenta es que la señal de error presente en el enlace provoque un movimiento mínimo de las estrellas dentro de la constelación, para que el receptor no cometa errores. Como es lógico, esto depende fundamentalmente de la SNR empleada.

Apéndice D

Transmisión de datos serie

A la hora de transmitir más de un bit de datos existen dos posibilidades: (1) enviar todos los bits a la vez utilizando tantos enlaces como bits o (2) enviar los bits uno detrás de otro. En el primer caso hablamos de **transmisión paralela** porque muchos bits llegan hasta el receptor a la vez, en paralelo,

mientras que en el segundo caso utilizamos el término **transmisión serie** para indicar que los bits llegan al receptor formando una hilera.

La transmisión paralela presenta la ventaja de que si existen hasta N líneas de transmisión, el tiempo de transmisión serie se divide entre N, aunque necesita N enlaces. Por este motivo se utiliza en distancias cortas como son los buses de datos de las computadoras, impresoras, etc.

La transmisión serie presenta la ventaja de que es N veces más barata en términos de enlace y por esto se emplea cuando las distancias que separan el emisor y el receptor son grandes, o cuando simplemente sólo existe un medio de transmisión (por ejemplo, un enlace por radio). En redes de computadoras, y en general, en cualquier red de transmisión de datos la forma más frecuente de transmisión es la serie.

Cuando hablamos de transmisión serie es también común emplear los términos de **transmisión síncrona** y **transmisión asíncrona**. Cuando se trata de enviar grandes cantidades de bits es normal agruparlos en bloques que faciliten su transmisión. Aparecen así los conceptos de carácter y trama de datos. Un carácter suele tener un tamaño de 8 bits y una trama de datos suele estar formada por una secuencia de caracteres, aunque también podemos encontrarnos casos donde los caracteres sean sólo de 7 bits y las

tramas de una longitud arbitraria.

Cuando los caracteres son transmitidos esporádicamente y además cada transmisión puede a priori realizarse en cualquier instante, se dice que la transmisión es asíncrona. Sin embargo, cuando los caracteres se transmiten sin pausa (formando una trama de datos) se habla de transmisión síncrona [8] de ese bloque de caracteres.

D.1. Sincronización de bits

Tanto para el caso de la transmisión serie como de la paralela, la determinación del instante de tiempo en que cada bit es recibido es una cuestión fundamental para su reconocimiento correcto. Observando la Figura B.1-(a), que muestra cómo sería la recepción de la componente de frecuencia fundamental de una señal digital en banda base que transporta la peor secuencia de bits transmitida a través de un enlace sin ruido, apreciaremos que el mejor momento para medir el valor de la señal recibida es justo en la mitad de la recepción de cada bit.

Para encontrar dichos instantes el emisor y el receptor utilizan sendos relojes. El del emisor indica cuándo enviar el siguiente bit y el del receptor cuándo medir el valor de la señal recibida. Ademas, para que la transmisión sea posible dichos relojes deben poseer la misma frecuencia, es decir, han de estar sincronizados. En una transmisión paralela el problema de la sincronización se resuelve enviando la señal producida por el reloj (la señal de reloj) a través de una línea diferente, dedicada. En el caso de la transmisión serie esto no es posible, y se emplean dos técnicas simultáneamente: (1) el envío previo de bits de sincronismo y (2) la utilización de señales que incor-

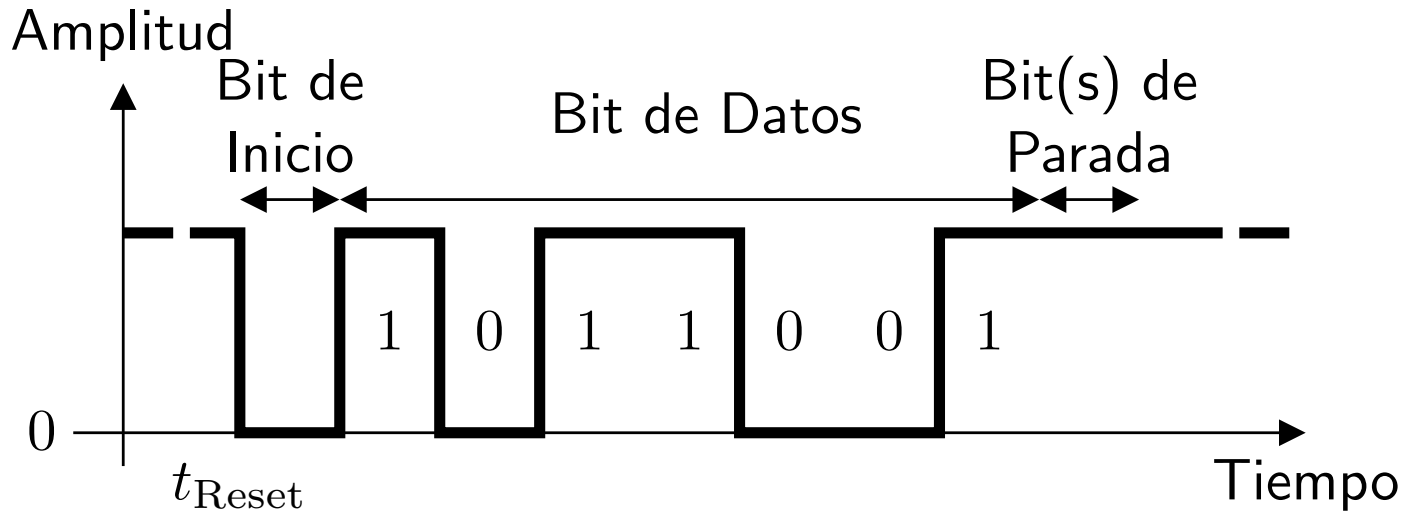


Figura D.1: Un ejemplo de transmisión serie de un carácter.

poran sincronía, es decir, que transportan junto con lo datos la información de sincronismo.

Los bits de sincronismo (que pueden ser en realidad sólo 1) anteceden a los bits de datos y sirven para que el receptor sincronice inicialmente su reloj. Por ejemplo, en la Figura D.1 se muestra una señal digital que transporta un pulso de sincronismo [3] y 7 bits de datos (un carácter). Se ha supuesto que cuando la línea está inactiva (no se transmite nada), el nivel de señal

recibido es 0, y que este es el nivel que representa también al bit 0. En la figura también aparece un bit de parada que representa el tiempo que la línea debe estar inactiva antes de transmitir un nuevo bit de inicio.

El bit de inicio indica que en el instante de tiempo t_{Reset} el reloj debe reiniciarse, y se ha supuesto que la precisión de los relojes del emisor y del receptor es suficiente como para que la pérdida de sincronismo no afecte a la recepción del séptimo y último bit de datos. Nótese que la duración del bit de inicio no puede utilizarse para ajustar el periodo del reloj porque el primer bit de datos no tiene por qué indicar cuándo finaliza el bit de inicio y comienza él (para comprender esto, basta con imaginar que en la Figura D.1 el primer bit de datos sea un 0).

El ejemplo de transmisión de un carácter que aparece en la Figura D.1 es también el ejemplo de una transmisión asíncrona. Señales de este tipo son las que se envían desde un teclado hasta la computadora cada vez que pulsamos una tecla. En este caso la fuente de datos genera caracteres en cualquier instante de tiempo y la secuencia de señalización mostrada en la Figura D.1 resulta la adecuada para su transmisión inmediata. Sin embargo, cuando el emisor genera secuencias de caracteres o en general secuencias de bits mucho más largas, el uso de bits de inicio y de parada insertados entre

los distintos caracteres supone un overhead considerable.

Para atajar este problema la mejor solución es enviar cada carácter uno a continuación del otro, sin insertar bits de parada y de sincronismo. Esto es justamente lo que se realiza en la transmisión síncrona que permite así tasas de transferencia superiores. El problema que esto plantea es que las secuencias de caracteres pueden ser arbitrariamente largas y sería necesario utilizar relojes extremadamente precisos para garantizar que la pérdida de sincronismo (que siempre existe,) no afecte a la recepción de los bits.

Como se dijo al comienzo de esta sección, para solventar la pérdida de sincronismo se utilizan señalizaciones con información de sincronía. En principio cualquier señal digital aporta cierta cantidad de información de este tipo, pero en algunos casos puede ser insuficiente. Por ejemplo, en la Figura D.1 la transmisión de bits contrarios indica dónde comienza y acaba cada bit, pero esto no ocurre así cuando se reciben bit iguales. Consecuentemente, los problemas de sincronización aparecen cuando recibimos largas cadenas de unos o de ceros y por este motivo se dice que la señalización utilizada en la Figura D.1, que se conoce como **señalización unipolar** [8] porque la señal siempre tiene el mismo signo, no aporta información de sincronía.

En el resto de esta sección estudiaremos otras formas diferentes de señal-

ización para datos digitales que poseen determinadas propiedades que las hacen adecuadas para contextos concretos. La primera de estas formas, la señalización unipolar, es utilizada cuando el enlace es capaz de transmitir energía en un único sentido[1] y cuando la combinación precisión del reloj versus longitud de la trama de datos no da lugar a desincronizaciones. Para aquellas situaciones donde esto no ocurre así se han ideado otras señalizaciones más eficientes.

[1] Ejemplos de enlaces que pueden hacer esto son los pares trenzados, las fibras ópticas y todos los enlaces mediante señales electromagnéticas.

D.2. Señalizaciones bipolares

Cuando se diseñan enlaces de transmisión de datos serie mediante cables eléctricos, una de las principales mejoras que podemos realizar de cara a reducir el coste consiste en utilizar un único hilo conductor en lugar de dos, como ocurre por ejemplo en el Telégrafo [21]. Cuando sólo tenemos como tensión de referencia la tierra local, la técnica para averiguar qué pulso es el que ha llegado (saber si se trata de un 0 o un 1) consiste en "estimar" la tensión de tierra que el emisor está utilizando, promediando el valor de la señal [22] que llega, ya que es bastante acertado suponer que durante la transmisión de grandes cantidades de datos, el número de bits iguales a 1 va a ser aproximadamente igual al número de bits iguales a 0. Si usamos una señal unipolar, la media de la señal recibida es siempre mayor que la señal que llegaría al receptor en ausencia de datos (línea inactiva o señal de tierra) y esto es un problema porque no sabríamos con certeza el nivel de tierra. Este inconveniente queda al menos parcialmente resuelto cuando usamos una señal eléctrica bipolar.

En la Figura D.2 se presenta la **señalización sin retorno a nivel cero** o **NRZL (Not Return to Zero Level)** que es la versión bipolar de la

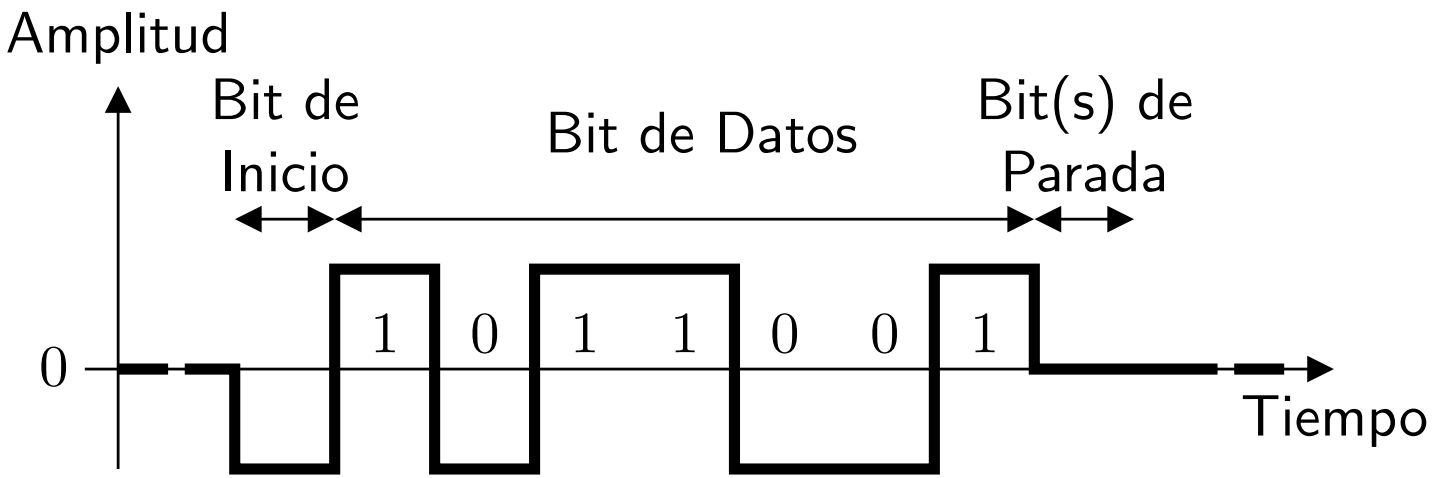


Figura D.2: Ejemplo de señalización NRZL.

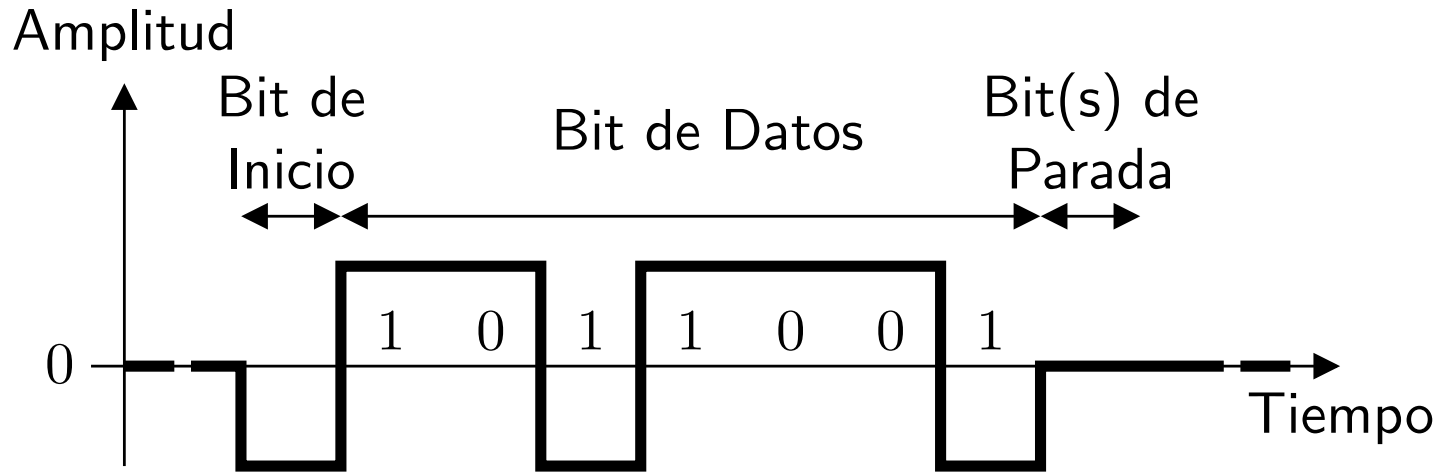


Figura D.3: Ejemplo de señalización NRZI.

señal mostrada en la Figura D.1. Como puede observarse, se trata de una versión idéntica a la señalización unipolar, excepto que la señal toma valores negativos: un 1 genera un pulso positivo y un 0 un pulso negativo. Nótese que NRZL posee los mismos problemas de sincronismo, la misma inmunidad frente al ruido y consume el mismo ancho de banda que la señalización unipolar.

D.3. Señalizaciones auto-reloj

En la Figura D.3 se muestra la **señalización sin retorno a nivel cero invertida** o **NRZI (Not Return to Zero Inverted)** que es la versión diferencial de la señalización NRZL. En este caso la señal sólo cambia cuando enviamos un 1, lo que aumenta las posibilidades de sincronización durante la transmisión de las secuencias de 1's. Esta señalización presenta la misma inmunidad frente al ruido y consume el mismo ancho de banda que NRZL pero, además, tiene la ventaja de que los terminales de pares trenzados se pueden conectar en cualquier orden porque el receptor distingue los 1's mediante cambios en la señal, independientemente de su polaridad. Por este motivo se dice que NRZI es una **señalización diferencial**.

En determinadas situaciones donde el ancho de banda no supone un problema, podemos usar la **señalización Manchester** que permite una sincronía constante del emisor y del receptor a través de la señal de datos. Consiste en enviar una transición $+A \rightarrow -A$ (donde A es el nivel de amplitud de la señal) cada vez que transmitimos un 0 y una transición $-A \rightarrow +A$ para un 1, tal y como se muestra en la Figura D.4. Ahora el ancho de banda requerido es exactamente el doble (en promedio) que para el caso de NRZI,

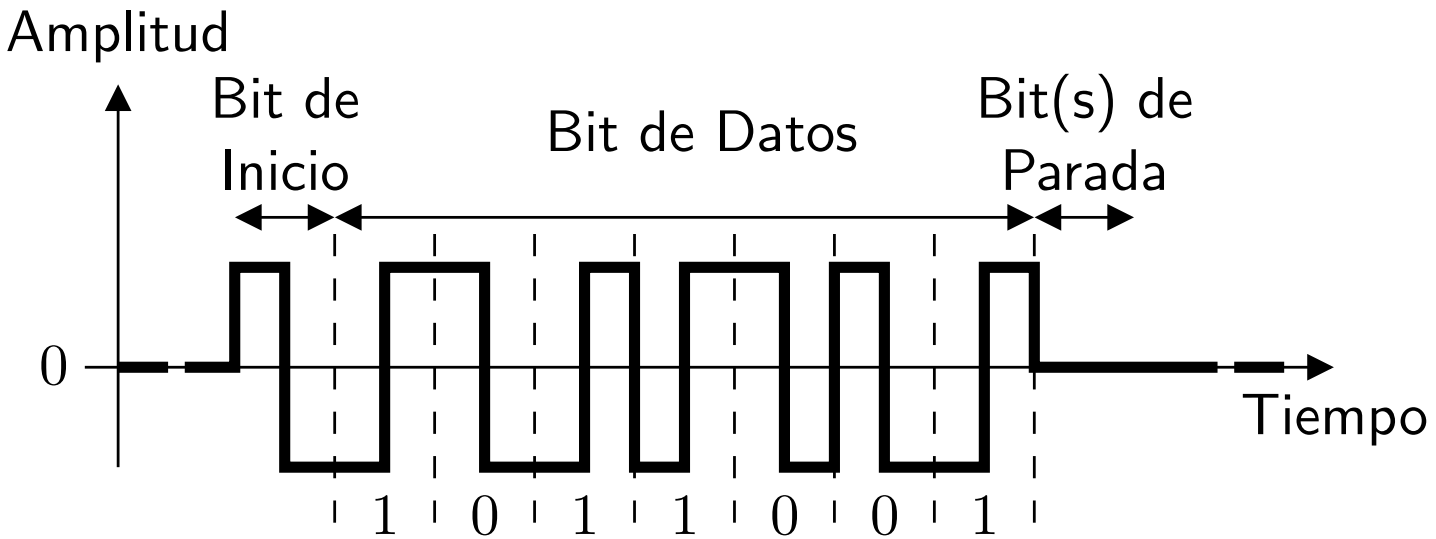


Figura D.4: Ejemplo de señalización Manchester.

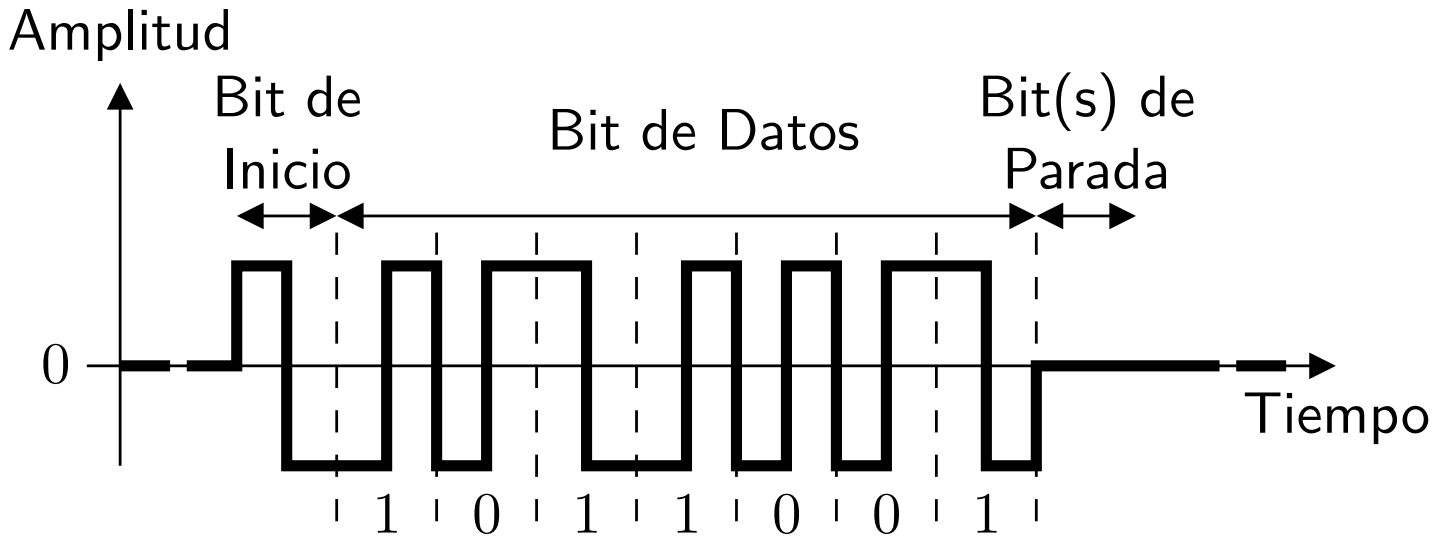


Figura D.5: Ejemplo de señalización Manchester diferencial.

NRZL y unipolar, aunque la inmunidad frente al ruido sigue siendo la misma. Nótese que ahora la media de la señal es exactamente igual a cero, siempre.

También existe una versión diferencial de esta señalización que se llama **señalización Manchester diferencial**. Consiste en provocar una transición al principio del intervalo si se transmite un 0 y de provocarla sólo en la mitad del intervalo en el caso de un 1 (véase la Figura D.5).

D.4. Señalizaciones más resistentes a errores

Todas las señalizaciones anteriores poseen una característica en común: son bi-nivel o bi-valuadas (sin contar con el nivel de línea inactiva). Esto les confiere una cierta capacidad para soportar el ruido de tipo blanco que está presente en el enlace, aunque su defensa es pobre cuando aparece ruido de tipo impulsivo que puede cambiar totalmente la polaridad de la señal.

Frente a este tipo de señales, existe otro gran grupo de señalizaciones que utiliza tres niveles de energía para representar la información digital y que son más resistente al ruido impulsivo. En concreto utilizan $+V$, $-V$ y 0. Por este motivo también se dice que este tipo de señales sigue una **señalización con retorno a zero** o **RZ (Return to Zero)** [8].

Una de las más simples se llama **señalización con inversión de marca alternada** o **AMI (Alternate Mark Inversion)** y consiste (vea la Figura D.6) en usar el nivel de ausencia de señal para señalizar un 0 y un nivel con polaridad alternante para señalizar un 1 [27]. Como puede verse, la media de la señal es cero y además cada 1 aporta información de sincronismo. El problema sigue existiendo con los 0's. Por otra parte, su resistencia al ruido radica en que para que un 0 se convierta en un 1 debe crearse un nivel de

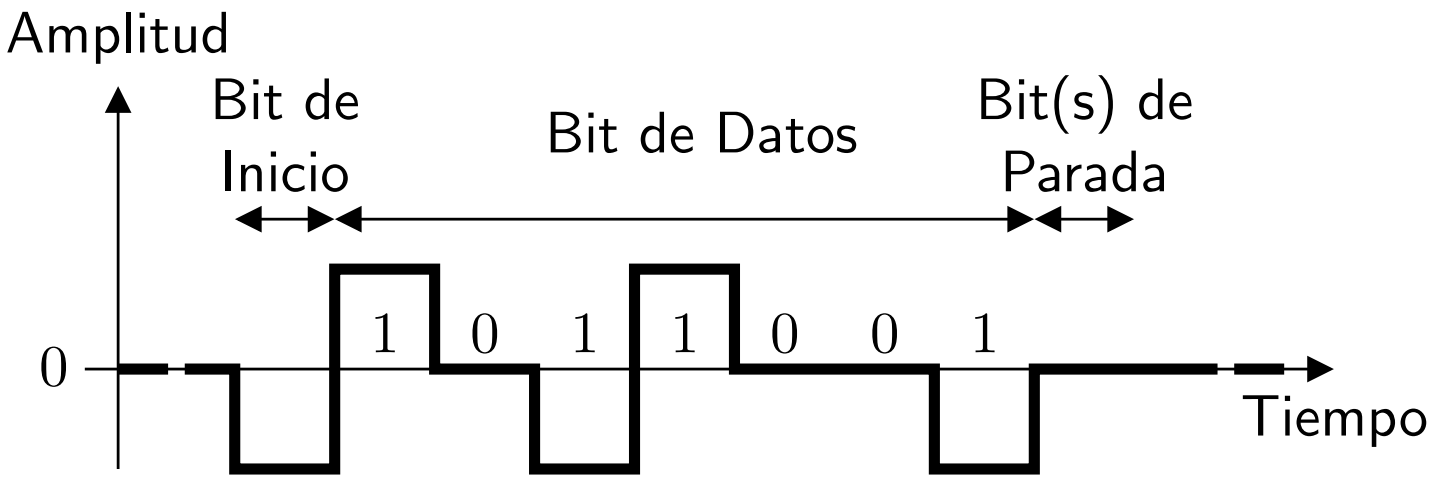


Figura D.6: Ejemplo de señalización AMI.

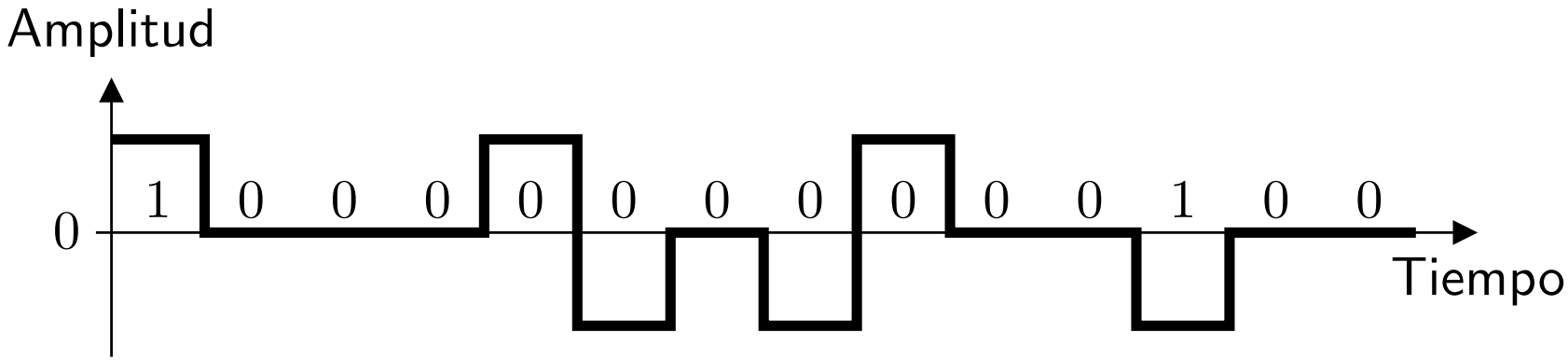


Figura D.7: Ejemplo de señalización B8ZS.

tensión suficiente y además cambiar la polaridad respecto del 1 anterior.

Con el fin de incorporar sincronismo también con los 0's, se diseñaron a partir de la AMI dos señalizaciones muy usadas actualmente: la **señalización B8ZS** y la **señalización HDB3**. Ambas se basan en la idea de romper la ausencia de transiciones que existe durante la transmisión de las secuencias de 0's, "violando" la propiedad de polaridad alternante que se define en AMI. En concreto, para el caso de B8ZS, cada secuencia 0000 0000 se sustituye por 000V B0VB donde una V significa "violación de la polaridad" y una B significa "polaridad correcta" respecto de la polaridad anterior [27, 8]. En

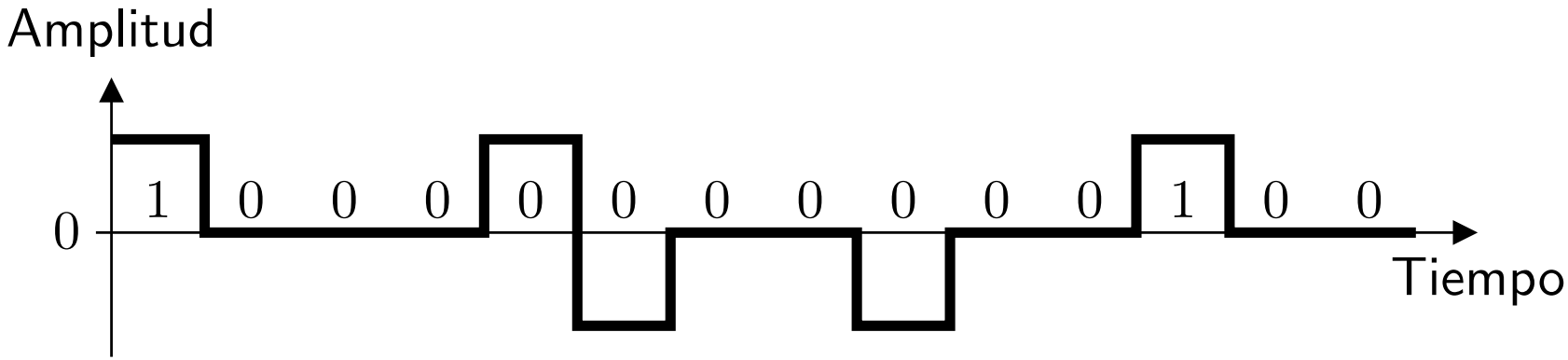


Figura D.8: Ejemplo de señalización HDB3. Se supone que el primer 1 es primero de la secuencia y por tanto hay un número impar de unos.

la Figura D.7 se muestra un ejemplo de aplicación. En el caso de HDB3 se sustituyen las secuencias de 4 ceros aplicando la siguiente regla: si el número de unos desde la última sustitución es par, entonces la secuencia 0000 se sustituye por B00V y si el número de unos desde la última sustitución es impar, la entonces la secuencia de 4 ceros se sustituye por 000V [8, 27]. Nótese que de nuevo, para calcular las polaridades de las sustituciones es necesario fijarse en la última polaridad utilizada. En la Figura D.8 hay un ejemplo.

D.5. Delimitación de tramas

Como se desprende de la sección anterior, la delimitación de las tramas de datos es un proceso necesario de cara a sincronizar el emisor y el receptor. Sin embargo, existen otras razones de importancia para limitar la longitud de las tramas que son: (1) la resistencia frente al ruido de transmisión y (2) la compartición de los medios. Como veremos, muchas de las técnicas de control de errores que se han desarrollado se basan en retransmitir aquellas tramas de datos que han llegado con errores. Si las tramas son muy largas, la probabilidad de que aparezcan errores aumenta de forma que no podríamos transmitir ninguna de ellas con éxito. Por otra parte, cuando las redes de transmisión utilizan medios compartidos y TDM (como ocurre por ejemplo en las redes Ethernet) no queda más remedio que limitar el tiempo de transmisión para acotar así el tiempo máximo de espera por parte de un emisor.

D.6. Transmisiones asíncronas

La manera en que las tramas son delimitadas y el tipo de señalizaciones utilizadas cambian si estamos empleando una transmisión asíncrona o síncrona. En una transmisión asíncrona la delimitación está implícita en la longitud finita (7 y 8 bits típicamente) y conocida tanto por el emisor como por el receptor de la trama de datos. Además, como las tramas son cortas no es necesario mantener sincronizados los relojes durante su transmisión, por lo que las señalizaciones NRZL y NRZI son las más utilizadas.

D.7. Transmisiones síncronas

Por el contrario, en el caso de la transmisión síncrona, donde un número en general variable de caracteres o bytes son transmitidos en cada trama, las formas de delimitación son mucho más sofisticadas y se utilizan señalizaciones con sincronía. El objetivo es evitar a toda costa un **error de delimitación de trama de datos** porque, para el caso concreto de una transmisión síncrona, implica la pérdida de grandes cantidades de bits como consecuencia de la des-sincronización entre el emisor y el receptor.

Lo mismo que ocurre en una transmisión de un carácter, para transmitir una trama de datos primero se envían una serie de pulsos de sincronismo que sirven para sincronizar los relojes. El uso de una serie de bits de sincronismo (en lugar de un único bit) no representa un overhead considerable ya que el ratio bits de datos / bits de sincronismo es muy alto. Por este motivo es corriente hablar de pulsos de sincronismo en plural (y en general al menos un carácter). Esto además suele hacer innecesario el uso de bits de parada ya que los bits de sincronismo suelen tener una duración suficiente para que el sistema pueda mantenerse sincronizado durante la transmisión de dos tramas consecutivas.

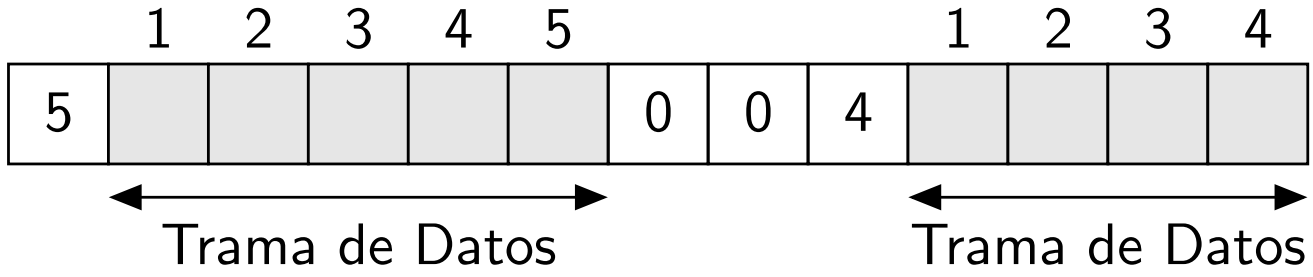


Figura D.9: Ejemplo de entramado usando recuento de caracteres. El primer carácter del marco indica su longitud en caracteres.

D.7.1. Recuento de caracteres

Como ya hemos dicho, la principal complicación a las que debe enfrentarse el diseñador de protocolos de transmisión serie síncrona a la hora de delimitar las tramas es que suelen ser de longitud variable. Ante esta situación, una de las técnicas más simples de "entramado" (framing en inglés) es el **recuento de caracteres**. Consiste en incluir al comienzo de cada trama un campo que indica cuántos caracteres contiene la trama. En la Figura D.9 tenemos un ejemplo sencillo donde se muestra cómo se delimitan dos tramas de datos. Aquí se puede apreciar además que es posible

usar la propia técnica de recuento de caracteres para mantener constantemente sincronizados el emisor y el receptor: cuando no se transmiten datos se envían constantemente tramas vacías. Este tipo de técnicas se utilizan normalmente en enlaces punto-a-punto, donde la línea de transmisión es dedicada. En enlaces multipunto esta técnica de sincronización continua es imposible debido a que existen múltiples emisores potenciales que no tienen por qué estar sincronizados entre sí. Además, esto supondría que el enlace está siempre ocupado lo que es inviable en enlaces multipunto.

El principal inconveniente del recuento de caracteres radica en que es bastante sensible al ruido de transmisión puesto que una alteración en un bit del campo que indica la longitud de la trama afecta a su longitud, lo que genera inmediatamente un error de delimitación de trama. Para salir de esta situación, el receptor debe buscar los bytes de sincronización inicial que precede a toda trama de datos y recuperar así la sincronización. Un ejemplo de protocolo de transmisión síncrona que utiliza el recuento de caracteres es el DDCMP (Digital Data Communication Message Protocol) [22].

Datos		Datos transmitidos
A C B	→	STX A B C ETX
A STX B C	→	STX A **DLE** STX B C ETX
A DLE B C	→	STX A **DLE** DLE B C ETX
DLE ETX	→	STX **DLE** DLE **DLE** ETX ETX

Figura D.10: Ejemplos de entramado usando relleno de caracteres. En negrita aparecen los caracteres de relleno.

D.7.2. Inserción de delimitadores

La alternativa al recuento de caracteres se llama **inserción de delimitadores**. La idea consiste en colocar delante y detrás de cada trama de datos un carácter especial (en general, una secuencia de bits especiales) que permitan identificar el comienzo y el fin de trama (recuérdese que las tramas no tienen por qué ir seguidas). Así, si se produce en error de delimitación de trama, la sincronización puede recuperarse buscando estos caracteres especiales a lo largo de las secuencias recibidas.

Una de las técnicas más simples (y más usadas) para la delimitación de tramas basada en la inserción de delimitadores es el **relleno de caracteres** (**character stuffing** en inglés) que se emplea en los protocolos serie síncronos BISYNC (BInary SYNchronous Communication) y PPP (Point to Point Protocol) [22]. El carácter que indica el comienzo de trama se denomina STX (Start of TeXt) y el que especifica el fin, ETX (End of TeXt). Hasta aquí todo perfecto, pero ¿qué ocurre si estos caracteres aparecen dentro de la trama de datos? Para evitar un error de delimitación de trama el emisor inserta delante de cualquiera de estos caracteres otro carácter especial llamado DLE (Data Link Escape) para indicar que la trama no acaba o comienza ahí. Entonces, ¿qué ocurre si el carácter DLE aparece dentro de la trama de datos? Para evitar que el receptor lo tome como un carácter especial el emisor lo precede de otro DLE. De esta forma, el receptor siempre que recibe un DLE lo ignora como dato y no atiende al significado de delimitación del siguiente carácter; sea lo que sea se trata de datos. En la Figura D.10 aparecen algunos ejemplos.

La principal desventaja del relleno de caracteres es que los datos deben organizarse en caracteres. Para evitar esta limitación existe una técnica semejante pero que delimita las tramas a nivel de bits. Por esto se llama

Datos		Datos transmitidos
0110 1010	→	0111 1110 0110 1010 0111 1110
0111 1110	→	0111 1110 0111 11**0**10 0111 1110
0111 1101	→	0111 1110 0111 11**0**01 0111 1110

Figura D.11: Ejemplos de entramado usando relleno de bits. En negrita aparecen los bits de relleno.

relleno de bits o **bit stuffing**. En concreto, la técnica de relleno que vamos a estudiar es la que se utiliza en el protocolo HDLC (High-level Data Link Control) [22].

En HDLC, el comienzo y el fin de cada trama se delimita mediante la secuencia 0111 1110. Cuando esta aparece entre los datos, el emisor la rompe insertando un 0 antes del último 1. En otras palabras, lo que el emisor hace es buscar y romper cualquier secuencia de más de 5 unos consecutivos. Así, el receptor no tiene problema para delimitar la trama porque sólo tiene que encontrar 6 unos consecutivos. De nuevo, ¿qué ocurre si la secuencia

0111 110 aparece en los datos? Como el receptor va a considerar el último bit como un bit de relleno, el emisor lo precede de un bit a 0. Así la recepción es correcta. En la Figura D.11 hay algunos ejemplos más.

El relleno de bits tiene dos grandes ventajas. La primera de ellas es que el overhead introducido por el protocolo de transmisión para delimitar las tramas es menor que para el caso del relleno de caracteres: el relleno genera sólo un bit de redundancia frente a ocho. Sin embargo, la mayor ventaja proviene de que existe la seguridad de que en ningún instante de la transmisión se emiten más de 6 unos consecutivos. Si repasamos las técnicas de señalización anteriormente presentadas vemos que una señalización NRZI donde en lugar de bascular con los unos basculemos con los ceros, o una señalización AMI, son especialmente convenientes para ser utilizadas con el relleno de bits ya que los problemas de sincronización quedan automáticamente resueltos.

Apéndice E

Códigos de corrección de errores

Los códigos de corrección de errores permiten localizar el error y corregirlo sin necesidad de retransmitir. Como ya hemos indicado, la clave está en

la introducción de la suficiente cantidad de redundancia. La idea es crear un código donde las palabras se diferencien en una cantidad de bits suficiente (dependiendo de la cantidad de bits erróneos que se desea corregir). A esta diferencia se le llama distancia de Hamming[1] [11]. Por ejemplo, en un código binario natural de n bits de longitud existen hasta 2^n palabras distintas y la distancia de Hamming es 1. Como puede verse, cuando la distancia es 1 entonces el código no permite detectar, ni aún menos corregir, ningún error, porque cualquier palabra con uno o más errores genera otra palabra del código.

La capacidad de detección y de corrección de errores de un código depende de su distancia de Hamming. En concreto, **para detectar al menos d errores, se necesita una distancia igual a $d + 1$**. Por ejemplo, el código 0000, 0011, 1100, 1111 tiene una distancia de Hamming igual a 2. Si se recibiera la palabra 1110, sabríamos que hay al menos un error y que las palabras que con más probabilidad fueron transmitidas son 1100 y 1111.[2] Nótese, sin embargo, que esta información es insuficiente para corregir el

[1] En honor al matemático estadounidense R.W. Hamming.

[2] Esto es así porque en general la probabilidad de que un único bit sea erróneo es mayor que la probabilidad de que dos o más bits sean erróneos.

error porque ambas palabras son equiprobables.

Sin embargo, **para corregir d errores, se necesita una distancia de $2d + 1$**. Por ejemplo, el código 00000 00000, 00000 11111, 11111 00000, 11111 11111, tiene una distancia igual a 5, por lo que puede llegar a corregir hasta 2 bits erróneos. Por ejemplo, si se recibe la palabra 00010 11011, lo más probable es que se tratara de la palabra 00000 11111. Obsérvese que nunca sabremos con certeza esto porque implicaría que sabemos que sólo dos bits iban a ser invertidos durante la transmisión. Este factor es el que provoca que ningún código de corrección de errores sea infalible.

Los códigos de corrección de errores son usados en tres situaciones muy concretas:

1. **Cuando la probabilidad de los errores de bit es alta**. En este caso utilizaremos además tramas muy cortas (1 o 2 caracteres a lo sumo).

   Ejemplo E.15: Considere un enlace que comete errores de transmisión aislados con una tasa de 10^{-3} errores/bit. Supóngase que la longitud de la trama de datos es de 1.000 bits. Esto significa que en promedio cada trama transmitida contendrá un bit erróneo y la transmisión utilizando

un código de detección de errores no sería posible.

2. **Cuando no es posible solicitar la retransmisión de datos.** Por ejemplo, en enlaces de tipo simplex no pueden solicitarse retransmisiones porque los datos sólo pueden ser enviados en un sentido. Por lo tanto, tenemos que tratar de blindar lo máximo posible los datos transmitidos.

3. **Cuando las latencias son muy altas**. Por ejemplo, en una transmisión desde una nave espacial hasta La Tierra es preferible utilizar códigos de corrección de errores a tener que indicar a la nave que repita una transmisión.

E.1. El código de Hamming

Hamming ideó además un sistema de codificación que permite recuperar un número de errores de transmisión por palabra transmitida arbitrariamente grande, aunque en esta sección se presenta (por motivos de simplicidad) el código que permite recuperar sólo un bit erróneo.

Para hacer esto, entre los bits de datos se insertan bits de paridad, concretamente en las posiciones 2^i siendo $i \geq 0$. Como puede deducirse fácilmente, si la trama contiene n de datos, entonces un código de Hamming introduce

$$\log_2(n+2) \text{ bits} \tag{E.1}$$

de redundancia.

Supongamos que el tamaño de los símbolos originales es de 7 bits. El código de Hamming inserta en las posiciones que son potencias de dos un bit de paridad (véase la Figura E.1).

Por ejemplo, el bit que está en la posición 1 (que tiene un único bit distinto de cero y que es el bit de menos peso) es el bit de paridad de los bits situados en las posiciones impares, porque todas ellas tienen el bit

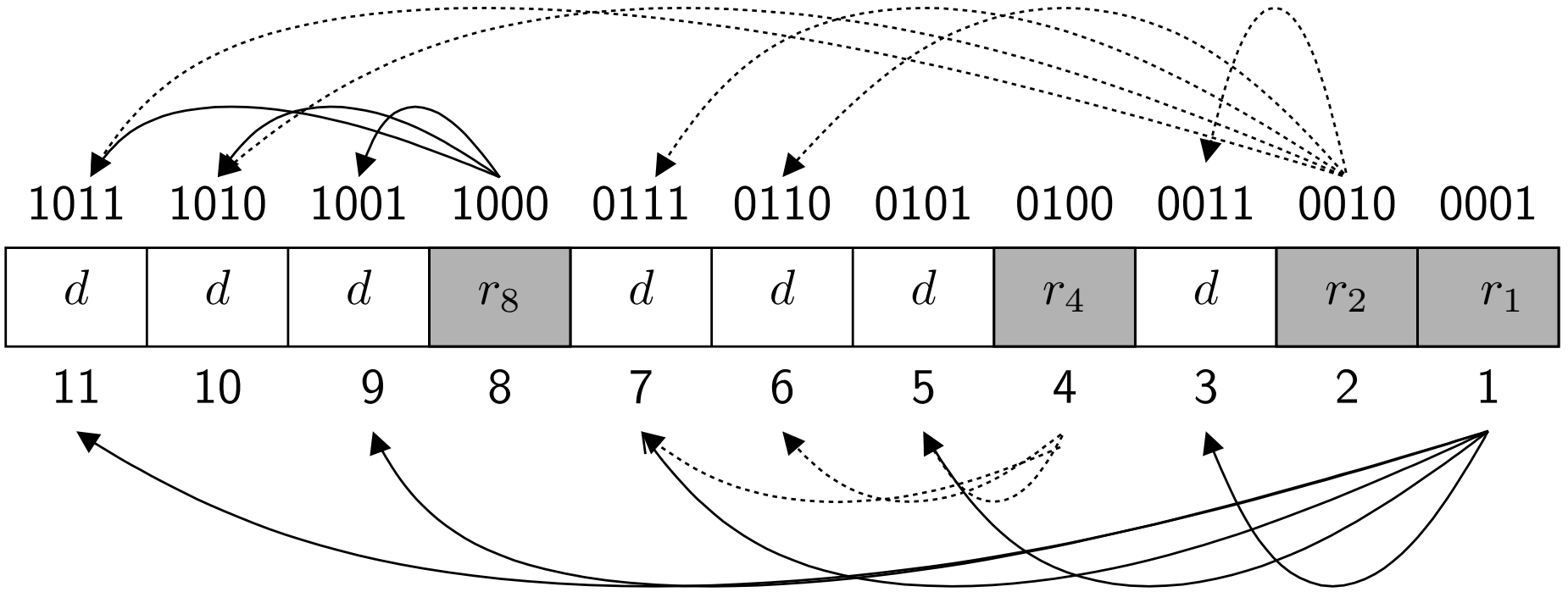
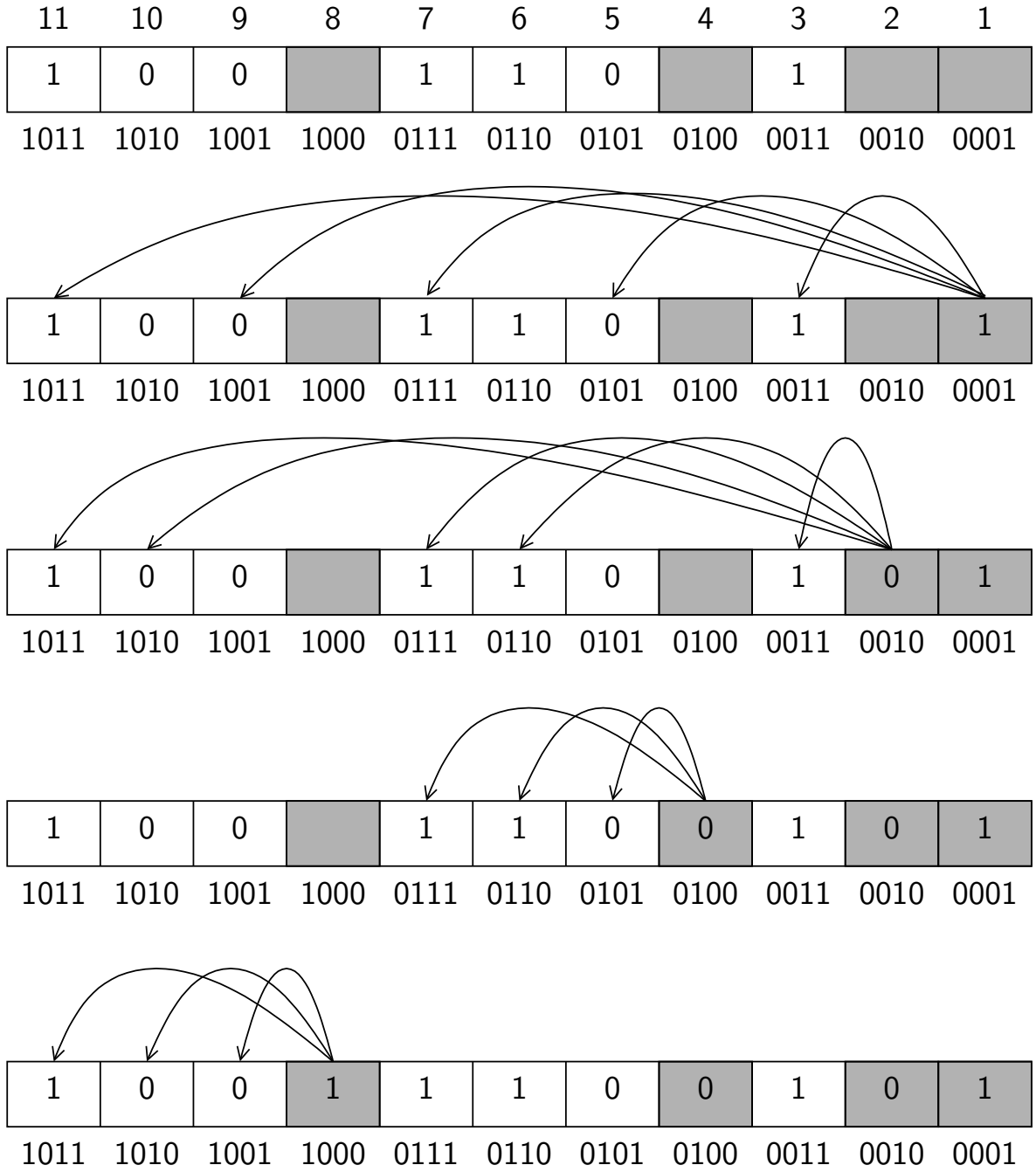


Figura E.1: Inserción de los bits de paridad en el código de Hamming. Los puntos seleccionados se corresponden con aquellos cuyo índice sólo tiene un bit a 1 y por tanto son potencias de 2.


11 10 9 8 7 6 5 4 3 2 1
1 0 0 1 1 0 1
1011 1010 1001 1000 0111 0110 0101 0100 0011 0010 0001
1 0 0 1 1 0 1 1
1011 1010 1001 1000 0111 0110 0101 0100 0011 0010 0001
1 0 0 1 1 0 1 0 1
1011 1010 1001 1000 0111 0110 0101 0100 0011 0010 0001
1 0 0 1 1 0 0 1 0 1
1011 1010 1001 1000 0111 0110 0101 0100 0011 0010 0001
1 0 0 1 1 1 0 0 1 0 1
1011 1010 1001 1000 0111 0110 0101 0100 0011 0010 0001

menos significativo igual a 1. En la Figura E.2 se muestra un ejemplo de codificación en el que se usa paridad par.

Para conocer si se ha producido un error de transmisión, el receptor recalcula los bits de paridad. Si no se ha producido ningún error, los bits de paridad deben coincidir con los que viajan con los datos. Si ha ocurrido un error, los bits de paridad que cambian indican dónde se ha producido el error. Por ejemplo, si el bit séptimo en la Figura E.2 cambiara de 1 a 0 durante una transmisión ruidosa, el receptor va a apreciar que los bits de paridad r_1, r_2 y r_4 son diferentes a los que se reciben. Como es más probable que se produzca un error en un bit que en tres, supondrá que el bit 7 (donde $7 = 1 + 2 + 4$) es el bit erróneo y lo corregirá. Este proceso se muestra también en la Figura E.3.

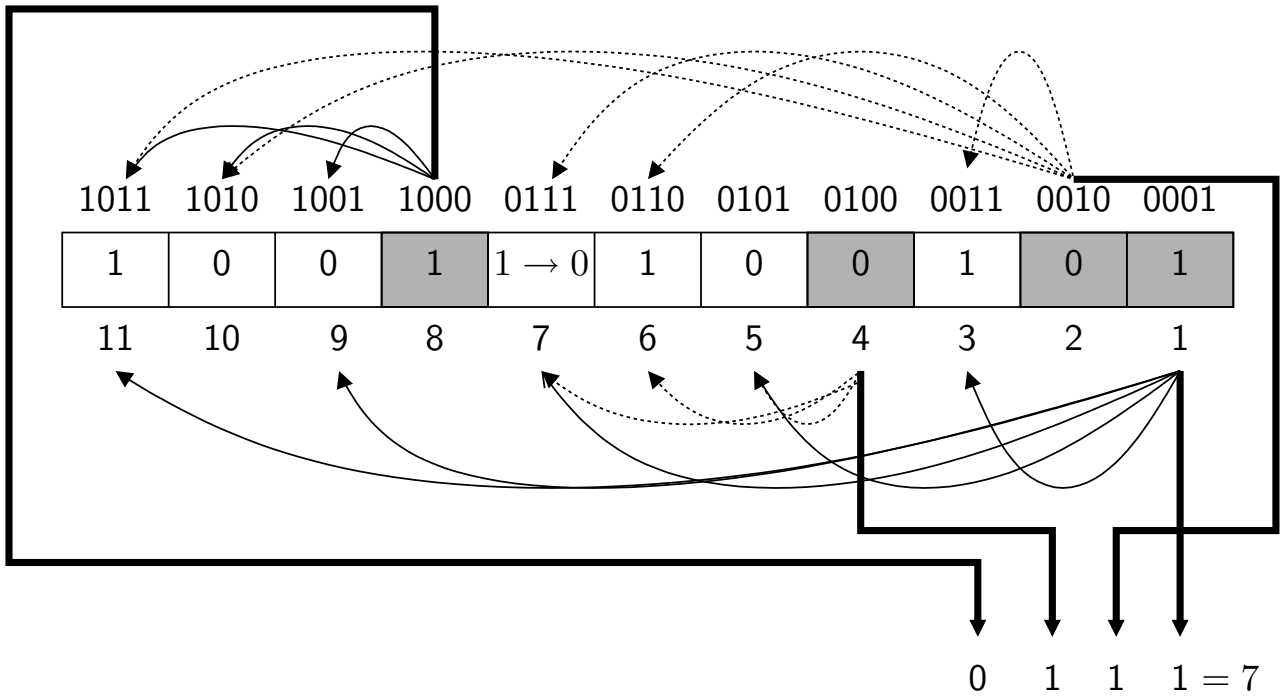


Figura E.3: Corrección de un error mediante el código de Hamming. Tras recalcular las paridades el número binario resultante indica la posición del bit erróneo. Un 1 en dicho código significa que la paridad es inorrecta, es decir, que el bit de paridad está bien calculado, y un 0 que es correcta. Por ejemplo, el bit de paridad r_1 está mal calculado (y por ese motivo colocalos un 1 en el bit de menos peso del código que indica dónde está el bit erroneo) porque según él, el número de 1's en las posiciones de índice impar es impar y sin embargo sale un número par.

Apéndice F

Fibras ópticas

El cable de fibra óptica es un medio guiado para la transmisión de luz (una forma de radiación electromagnética). En general se utiliza para transmitir señales digitales mediante el método de encender y apagar una fuente de luz [8]. En cierta medida este proceso es básicamente una modulación

ASK cuando un nivel de tensión es el que define la aparición o desaparición de la señal portadora (la onda de luz), cuya frecuencia va a depender del color de la luz. Si es luz monocromática entonces la portadora utiliza una única frecuencia, lo que suele ser lo más común porque el uso de una portadora monocromática hace posible el uso de una multiplexación en frecuencia en la fibra óptica. Esta técnica de multiplexación se llama **multiplexación por división en longitudes de onda** o WDM (Wavelength-Division Multiplexing) [27] que consiste en la utilización de colores distintos. En 1997 se alcanzó un hito cuando en los Laboratorios Bell se demostró que una fibra óptica es capaz de transmitir 100 haces de 10 Gbps proporcionando una capacidad total de 1 Tbps. Esto puede darnos una idea de la altísima capacidad que poseen los enlaces de fibra óptica.

La luz es una onda electromagnética, es decir, se trata de la combinación de un campo eléctrico y otro magnético vibrando en direcciones perpendiculares, que a su vez son perpendiculares a la dirección de propagación de la onda. La velocidad de propagación de la luz, y más en general de cualquier onda electromagnética, depende de parámetros que caracterizan electromagnéticamente el medio (permeabilidad y susceptibilidad). En el vacío la velocidad de propagación es de casi 300.000 km/s y en las fibras ópticas es

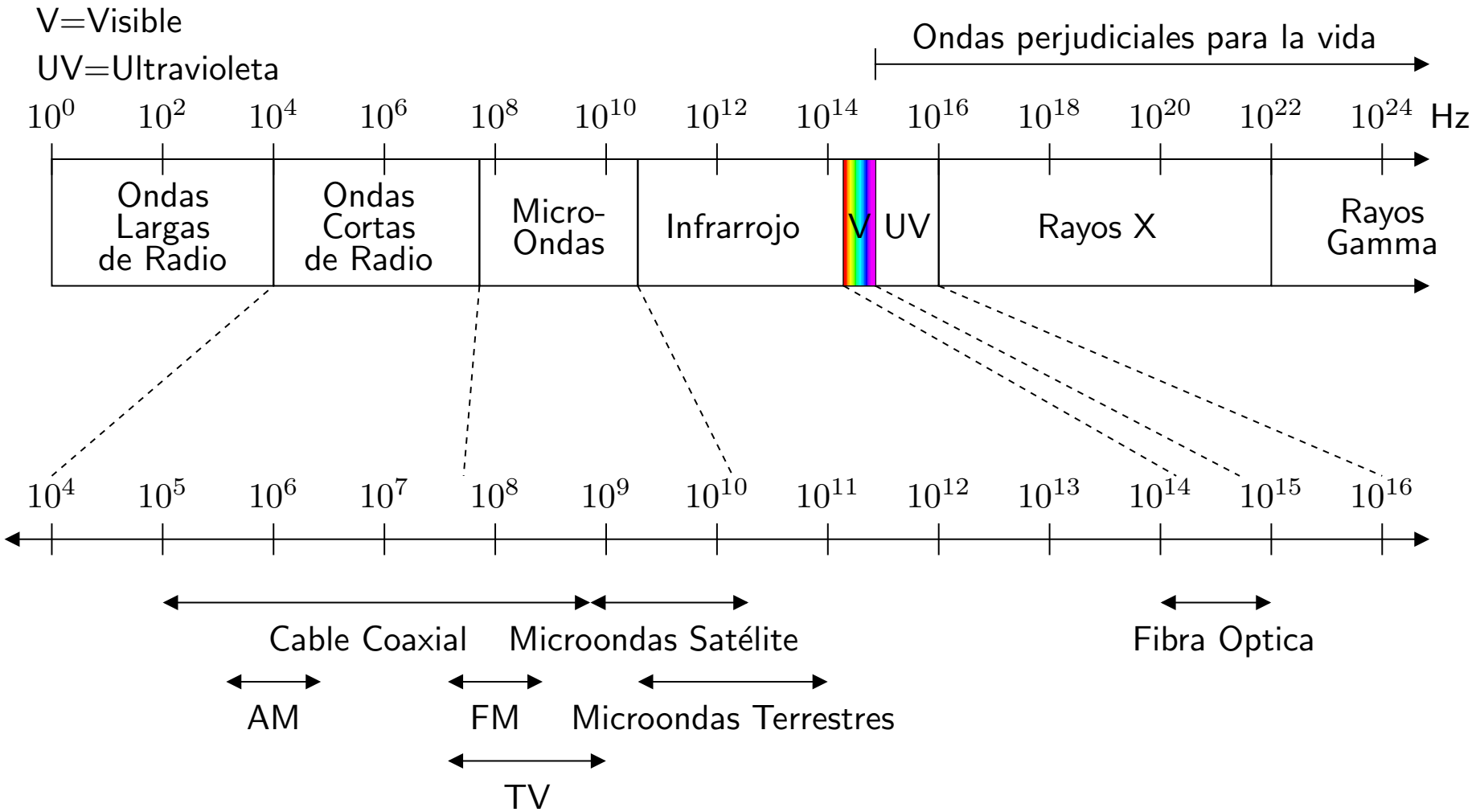


Figura F.1: El espectro electromagnético usado en telecomunicaciones [27, 32]. Nótese que la escala es logarítmica.

de 200.000 km/s. En la Figura F.1 se muestra el espectro electromagnético, donde se puede observar que la luz, como onda que es, ocupa la banda que va desde el infrarrojo hasta el ultravioleta.

En un medio transparente y homogéneo (cristal, plástico, agua, aire, etc.) la luz se propaga en línea recta y a una velocidad constante. Sin embargo, si en su propagación la luz pasa de un medio a otro (con diferentes niveles de permeabilidad y susceptibilidad), su velocidad y dirección de propagación cambian abruptamente. Este cambio en la dirección de propagación provocado por dicha modificación de los índices de permeabilidad y susceptibilidad electromagnética de se le conoce como **refracción**.

Los valores de la permeabilidad y la susceptibilidad son características del medio, a partir de los cuales se define la densidad del medio (electromagnéticamente hablando) y de los cuales depende el valor de la velocidad de propagación de una onda electromagnética en dicho medio. Cuando la luz pasa de un medio menos denso a un medio más denso (electrómagneticamente hablando), por ejemplo, del aire al agua, el ángulo de incidencia i es mayor que el ángulo de refracción r. Sin embargo, cuando pasa de un medio más denso a uno menos denso ocurre al contrario, es decir, el ángulo de incidencia i es menor que el ángulo de refracción r (véase la Figura F.2).

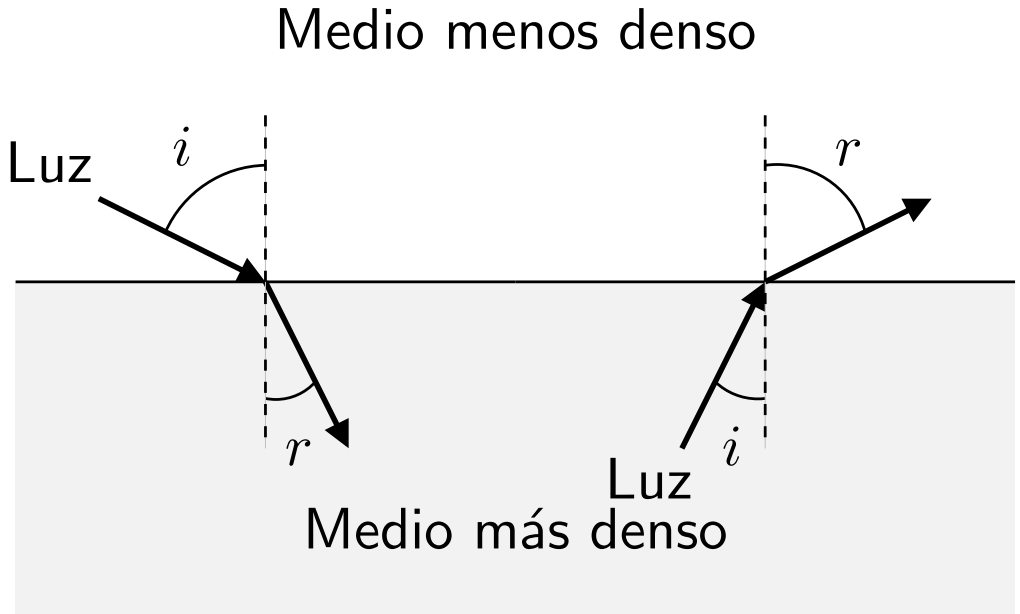


Figura F.2: Fenómeno de refracción de la luz producido al incidir sobre la superficie de separación de dos medios con diferentes densidades.

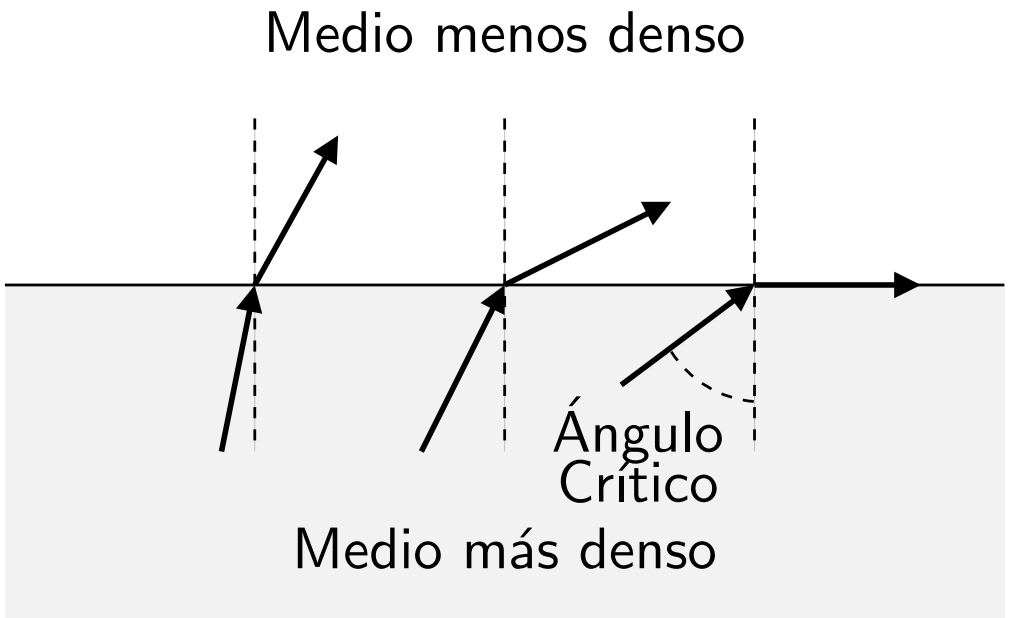


Figura F.3: Ángulo de incidencia crítico en la refracción de la luz.

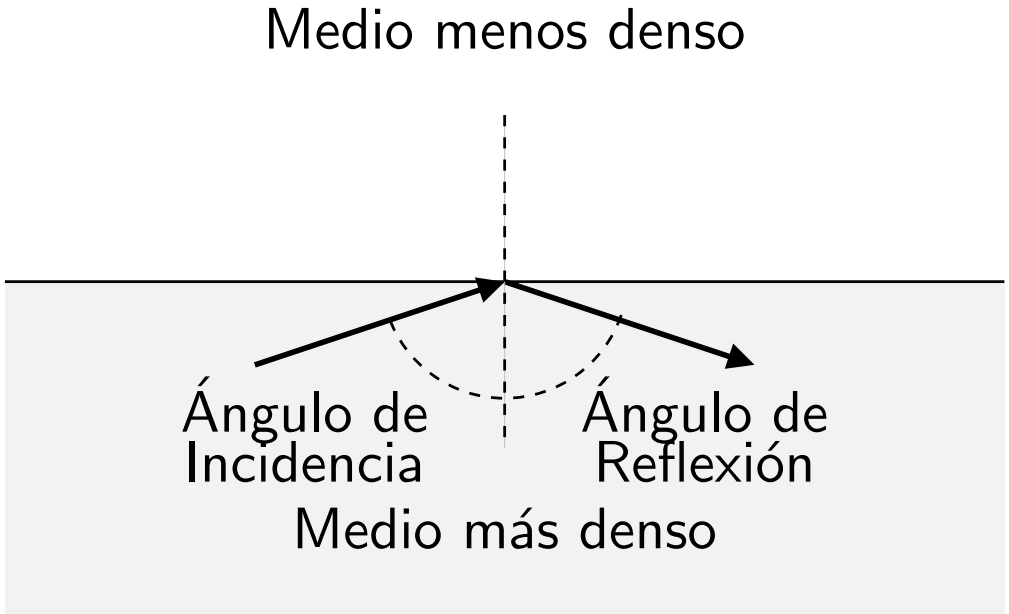


Figura F.4: Reflexión de la luz al incidir sobre la superficie de separación de dos medios con diferentes densidades.

Cuando la luz pasa de un medio más denso a otro menos denso, existe un ángulo de incidencia i para el cual la luz no se refracta ($r = 90^o$), sino que se propaga de forma paralela al plano de separación de ambos medios (véase la Figura F.3). A este ángulo se le llama **ángulo de incidencia crítico**.

Cuando la luz atraviesa medios de diferentes densidades, no toda la

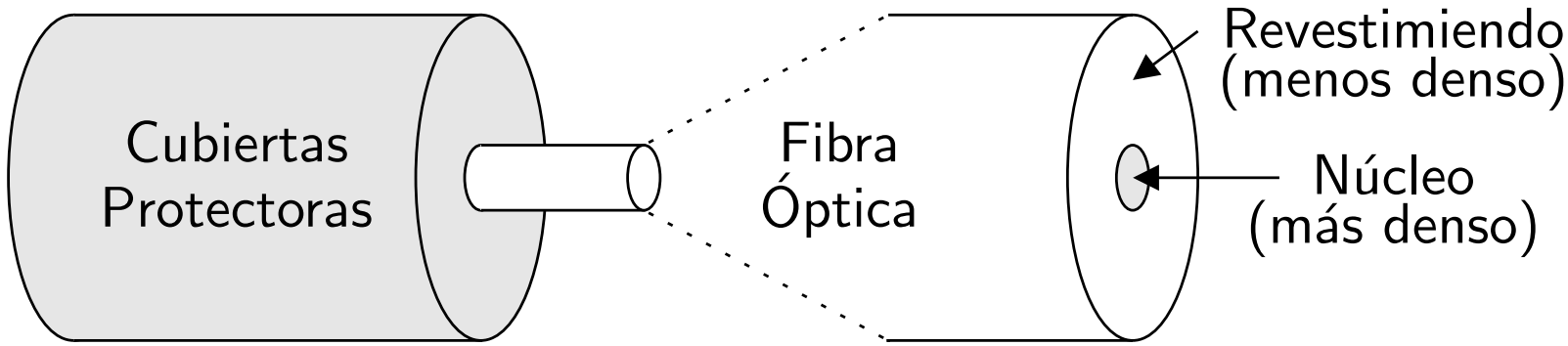


Figura F.5: Esquema de una fibra óptica. El núcleo y el revestimiento son materiales transparentes aunque con diferente densidad. La cubierta protectora suele estar formada por varios materiales que protegen la fibra de la luz externa y de los golpes.

energía consigue pasar al otro medio, sino que una cierta cantidad se refleja y continua transmitiéndose en el mismo medio de donde proviene. A este fenómeno se le denomina **reflexión**. En este caso, el ángulo de incidencia i y el ángulo de reflexión son idénticos (véase la Figura F.4).

Las fibras ópticas aprovechan el fenómeno de la reflexión de la luz para guiar (transportar en línea no recta) señales luminosas a grandes distancias.

Dependiendo de la forma en que se construye la fibra y de la forma en la que se inyecta la luz dentro de la fibra, existen diferentes modos de propagación y capacidades de transmisión. La base de todas las fibras ópticas es la construcción de un tubo o cilindro de material transparente (normalmente cristal o plástico) muy delgado que posee al menos dos densidades diferentes (véase la Figura F.5). En el centro o núcleo se encuentra el medio más denso (el que más lentamente transmite la luz) y en el revestimiento se encuentra el medio menos denso. Por tanto, la luz va a reflejarse en el revestimiento y a quedar confinada en el núcleo. Externamente, la fibra se recubre de cubiertas que impiden que la luz externa, la humedad o los aplastamientos alcancen la fibra y que la luz interna alcance el exterior por refracción.

Las fibras ópticas más sencillas son las llamadas **multimodo de índice discreto** y se construyen de forma que el núcleo y el revestimiento tienen densidades diferentes aunque constantes. Si colocamos una fuente de luz en un extremo de la fibra, cierta cantidad de los rayos penetran en el revestimiento y en el núcleo. Los haces que entran al revestimiento son absorbidos por la capa externa opaca, sin embargo, otros consiguen entrar en el núcleo. Dado que la haz incidente alcanza la fibra óptica con diferentes ángulos de incidencia, en la fibra óptica encontraremos que parte de dicha energía se re-

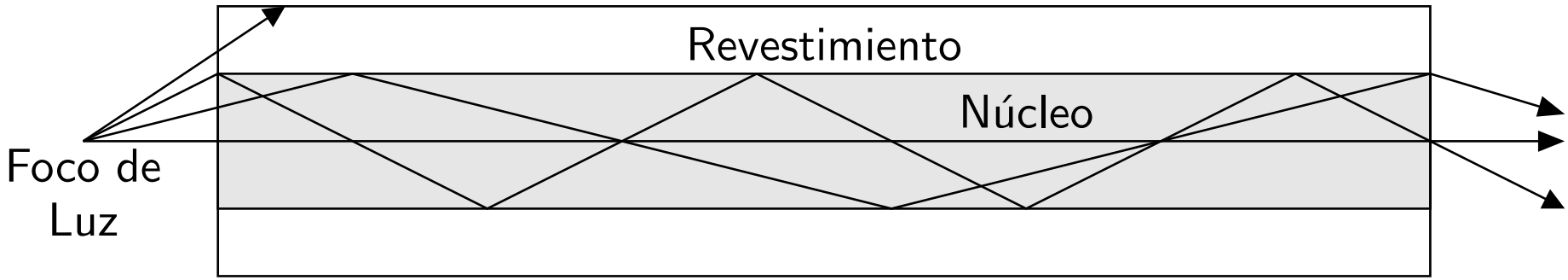


Figura F.6: Transmisión en una fibra óptica multimodo de índice discreto.

fractará en el revestimiento y será absorvida por la capa externa opaca; otra parte será reflejada con un ángulo de reflexión igual al de incidencia y otra parte se propagará de forma paralela al revestimiento (aquellos haces que indicen con un ángulo de incidencia igual al ángulo crítico). Véase la Figura F.6. Por tanto, dentro del núcleo existen rayos que se propagan sin reflejarse (rayos axiales [27]) y otros que sufren un número de reflexiones que depende del ángulo de incidencia de entrada en el núcleo y de la longitud de la fibra. Puesto que la densidad del núcleo es constante, la velocidad de propagación de la luz a través de él también lo es. Como consecuencia, los haces que más

distancia recorren alcanzan más tarde el otro extremo de la fibra. Debido a que la señal recibida en el otro extremo de la fibra es la suma de todas las señales que han recorrido diferentes caminos a través de la fibra óptica, la anchura de los pulsos de luz aumenta conforme se propagan, es decir, se ha producido una **distorsión por retardo** Esta perturbación constituye el principal factor que limita la capacidad de la fibra óptica, porque la separación entre símbolos se dificulta conforme aumenta la tasa de transmisión.

Para resolver este problema hay que evitar que los diferentes haces de luz que consiguen entrar en el núcleo en el mismo instante de tiempo alcancen el otro extremo en instantes diferentes. Para ello se han propuesto tres soluciones diferentes.

La primera de ellas consiste en utilizar fuentes de luz que producen frentes de onda planos, es decir, haces de luz que se propaguen en una única dirección, y esto es justamente lo que hace un laser (Light Amplification by Stimulated Emission of Radiation)[1]. En lugar de utilizar LEDs (Light Emitting Diodes) se usan ILDs (Injection Laser Diodes).

[1] En un laser todos los haces de luz se propagan en una única dirección y la onda generada es monocromática, es decir, de una única frecuencia (color) y fase.

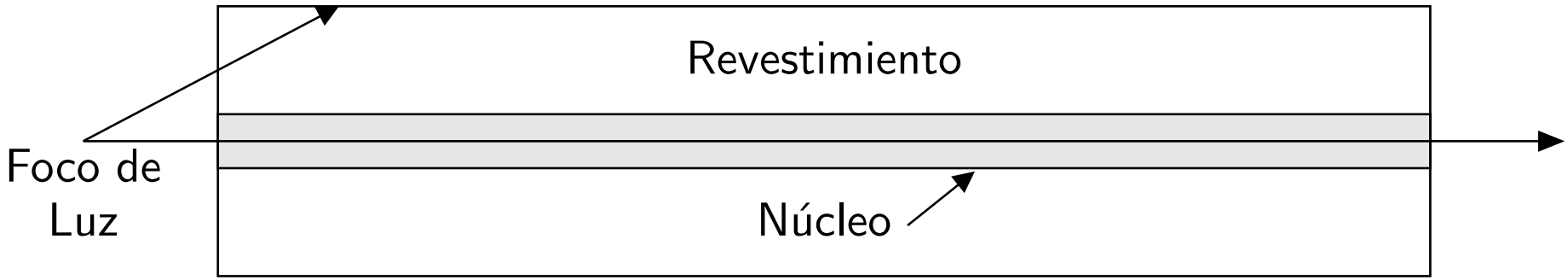


Figura F.7: Transmisión en una fibra óptica monomodo.

La segunda solución consiste en reducir el diámetro del núcleo hasta el punto de forzar a que todo rayo que consiga entrar en él tenga un ángulo de incidencia tan grande que sea crítico y por lo tanto sólo el rayo axial se propaga (véase la Figura F.7). A este tipo de fibras se les llama **fibras ópticas monomodo**. Las fibras monomodo son las más eficientes, pero también las más caras.

La tercera solución se basa en construir una fibra con un gradiente decreciente de densidad desde el núcleo hasta el revestimiento (véase la Figura F.8). Ahora, aquellos rayos de luz que no son axiales aceleran proporcional-

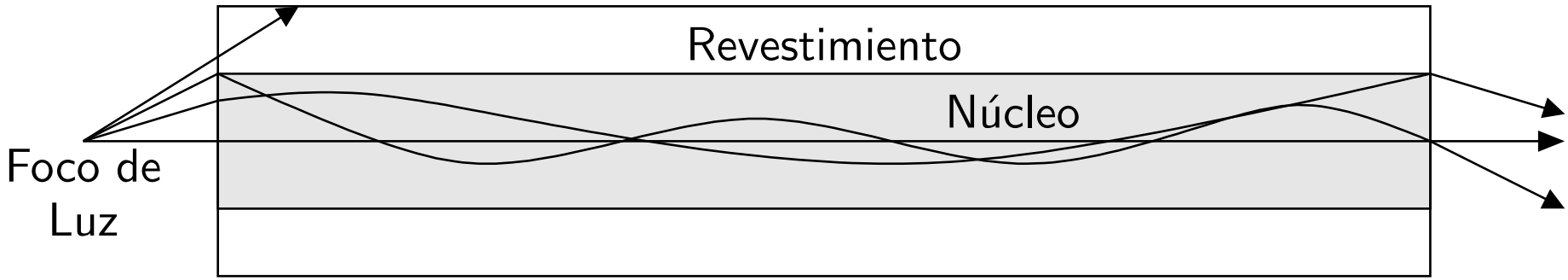


Figura F.8: Transmisión en una fibra óptica multimodo de índice gradual.

mente a la desviación de la dirección de propagación del rayo axial. La idea es permitir que los haces que más distancia recorren sean también los que más velozmente se propaguen. De esta forma se reduce la dispersión. A este tipo de fibra se le llama **fibras ópticas multimodo de índice gradual**.

Las principales ventajas de las fibras ópticas son:

1. **Una elevadísima capacidad**.
   Un canal de fibra tiene un ancho de banda típico de $10^{15} - 10^{14} = 0,9 \times 10^{15}$ Hz (véase la Figura F.1). Por tanto, simplemente codificando 1 bit/elemento de señalización la capacidad estimada de Nyquist

Cuadro F.1: Capacidades típicas de las fibras ópticas [22].

Fibra Óptica	Capacidad	Distancia
Multimodo	100 Mbps	2 km
Monomodo	$100 - 2.400$ Mbps	40 km

es de $1,8 \times 10^{15}$ bits/segundo o lo que es lo mismo, 1.800 Tbps. De hecho, la principal limitación de los sistemas de transmisión basados en fibra óptica la constituyen los elementos de transducción que transforman las señales luminosas en eléctricas y viceversa.

2. **Unas tasas de ruido muy bajas**.
   La fibra óptica es inmune a los campos electromagnéticos externos porque éstos no afectan a la luz (excepto la propia luz externa que no es problema gracias a la capa externa opaca). Como las fibras no irradian energía, tampoco existen problemas de diafonía y muchas fibras pueden colocarse en paralelo porque además son muy ligeras.

3. **Baja atenuación de las señales**.
   Esto permite instalar los repetidores mucho más separados que en el caso del par trenzado o el cable coaxial. En la Tabla F.1 se muestra la capacidad en función de la distancia de algunos tipos de fibras ópticas.

Apéndice G

Enlaces de radio

Las ondas de radio son emitidas mediante antenas, que simplemente son conductores por los que circula una corriente eléctrica de una cierta frecuencia, en general, no superior a 1 GHz para que sean consideradas como ondas de radio. Cuanto mayor es la frecuencia, más pequeña puede

ser la antena. Las ondas de radio se transmiten en todas las direcciones del espacio (son omnidireccionales) lo que las hace ideales para transmisiones de tipo broadcast.

Las ondas de radio, como ondas electromagnéticas que son, se propagan en el vacío a una velocidad de casi 300.000 km/s y son capaces de atravesar otros medios, como el agua o las paredes. En general, la absorción (o atenuación) que sufren las ondas de radio en un determinado medio dependerá de las propiedades electromagnéticas de dicho medio, así como de la frecuencia de las ondas. Es conocido que las ondas de radio de longitudes de onda mayores (frecuencias más bajas) se utilizan para transmisiones a larga distancia y para comunicarse, por ejemplo, con los submarinos sumergidos. De hecho, las ondas de radio de longitudes de onda más largas (frecuencias más bajas) pueden transmitirse a cientos de kilómetros y no se propagan en línea recta, sino que siguen la curvatura de La Tierra [27]. Al aumentar la frecuencia, este efecto desaparece pero las ondas se reflejan en la ionosfera[1], por lo que las distancias recorridas son también enormes. De hecho, en días

[1] La ionosfera es la capa de partículas cargadas que rodea La Tierra a una altura de 100 a 500 km.

propicios un radio-aficionado es capaz de escuchar su propia voz como un eco cuando es su señal la que da la vuelta a La Tierra y le llega de nuevo al emisor.

Los enlaces de radio no son normalmente utilizados para diseñar redes de computadoras de área amplia fundamentalmente por dos motivos:

1. Las ondas de radio son omnidireccionales, alcanzan largas distancias y atraviesan objetos (como las paredes de las casas). Bajo estas circunstancias, el medio de transmisión tiene una topología lógica de bus, donde mientras una estación escribe sobre el medio, el resto deben escuchar. La única solución a este problema es usar multiplexación (en el tiempo o en la frecuencia). Por ejemplo, las emisoras de radio comerciales utilizan FM (Frequency Modulation) y AM (Amplitude Modulation) para transmitir. El espectro electromagnético (véase la Figura F.1) se divide en bandas iguales y a cada emisora se le asocia uno de ellos.

2. El ancho de banda del espectro de frecuencias usado para transmitir señales de radio es insuficiente para acomodar grandes velocidades de transmisión. Por ejemplo, cuando un submarino está sumergido sólo

puede comunicarse con tierra a través de ondas de radio de muy baja frecuencia, por lo que las comunicaciones son muy lentas.

Para aumentar la capacidad total del medio de transmisión hay que usar frecuencias altas y bajas potencias de emisión. Esto es lo que ha permitido la aparición de la telefonía móvil y de sistemas de redes de área local para transmisión de datos entre computadoras. En el primer caso, la superficie terrestre se divide en celdas (de ahí el nombre también de telefonía celular) de igual tamaño, que pueden ser modeladas como hexágonos (aunque en realidad se parecen más a circulos). En cada uno de ellos se usan frecuencias que no se utilizan en ninguna de las celdas colindantes, lo que significa que pueden utilizarse de nuevo a dos celdas de distancia (véase la Figura G.1). Un teléfono móvil tiene cobertura sólo si está en contacto con una de las estaciones receptoras situadas (normalmente) en el centro de cada celda. Cuando el usuario conversa, se usa una frecuencia para transmitir y otra para recibir, y si se mueve desde una celda a otra, ambas frecuencias son diferentes a las anteriores y diferentes a cualquiera de las que está usando otro usuario en esa celda.

En los últimos años los enlaces inalámbricos han sufrido un avance gigan-

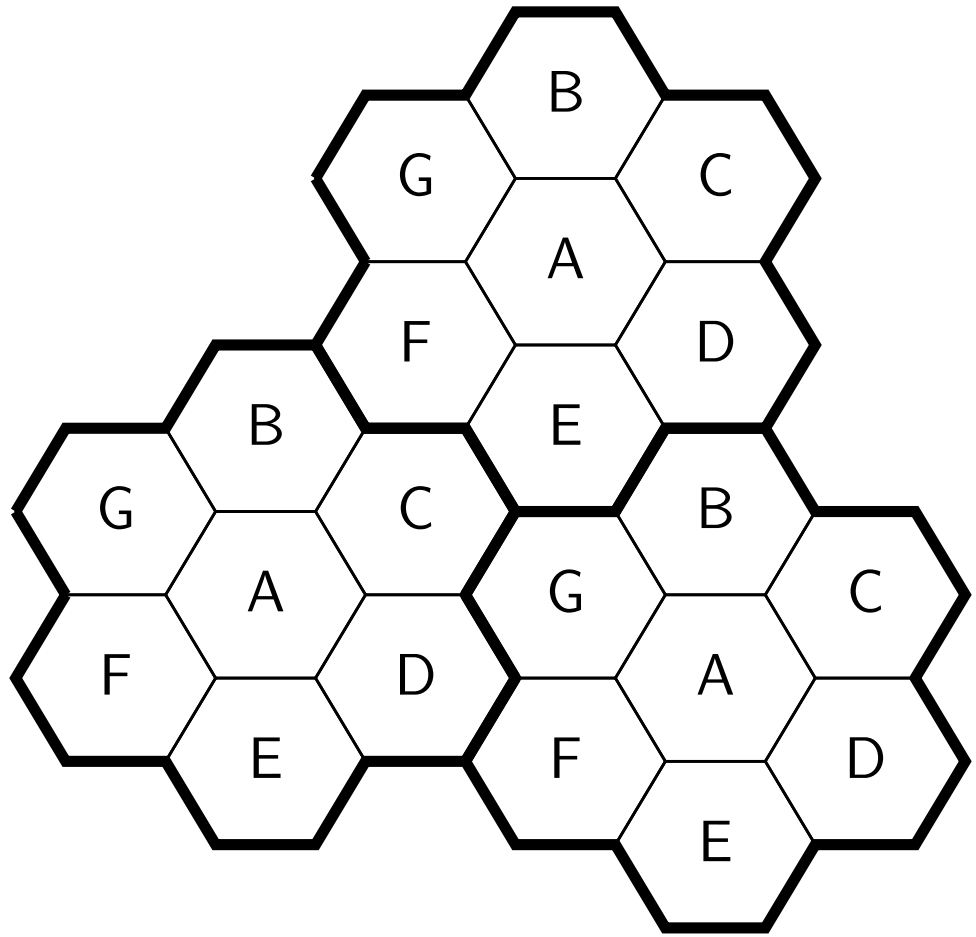


Figura G.1: Utilización de las frecuencias en telefonía móvil (celular) [32]. Cada letra es un conjunto de bandas de frecuencia igual al número máximo de personas que pueden hablar en cada celda.

tesco debido al desarrollo de la telefonía móvil digital. En USA y Canadá se usa una norma llamada PCS (Personal Communication Services). En el resto del mundo, GMS (Global Mobile System) es el estándar de telefonía digital sin cables. Como era de esperar, ambos son incompatibles [32].

Apéndice H

Enlaces de microondas

Aquellas ondas electromagnéticas comprendidas aproximadamente entre 1 GHz y 100 GHz son consideradas como microondas (véase la Figura F.1). Las microondas son mucho más direccionales que las ondas de radio aunque siguen desviándose (por suerte) con la curvatura de La Tierra [27]. Esto per-

mite distanciar las antenas a nivel terrestre algo más que si esta desviación no se produjera.

Debido a su alta direccionalidad se pueden diseñar antenas (normalmente parabólicas) que generan haces muy focalizados, por lo que sobre una misma localidad geográfica es posible establecer muchos enlaces punto a punto[1]. Además, como la anchura del espectro de las microondas es muy superior al de las ondas de radio, los enlaces de microondas son muy usados para diseñar enlaces con altas capacidades a larga distancia. El único problema a este respecto es que las microondas sólo se propagan sin atenuarse cuando lo hacen a través del aire y del vacío. De hecho, la lluvia puede llegar a ser un gran problema porque absorbe las microondas[2]. Por tanto, para enlaces de muy larga distancia (más de un centenar de kilómetros) son necesarias estaciones de repetición. Si las distancias a recorrer son muy grandes, suele

[1] En las torres de los teléfonos móviles es frecuente encontrar antenas parabólicas que emiten microondas. Estas constituyen los enlaces que ponen en contacto a los usuarios con el resto de la red telefónica.

[2] Como ya sabemos, los hornos de microondas calientan los alimentos porque estos tienen un cierto contenido de agua.

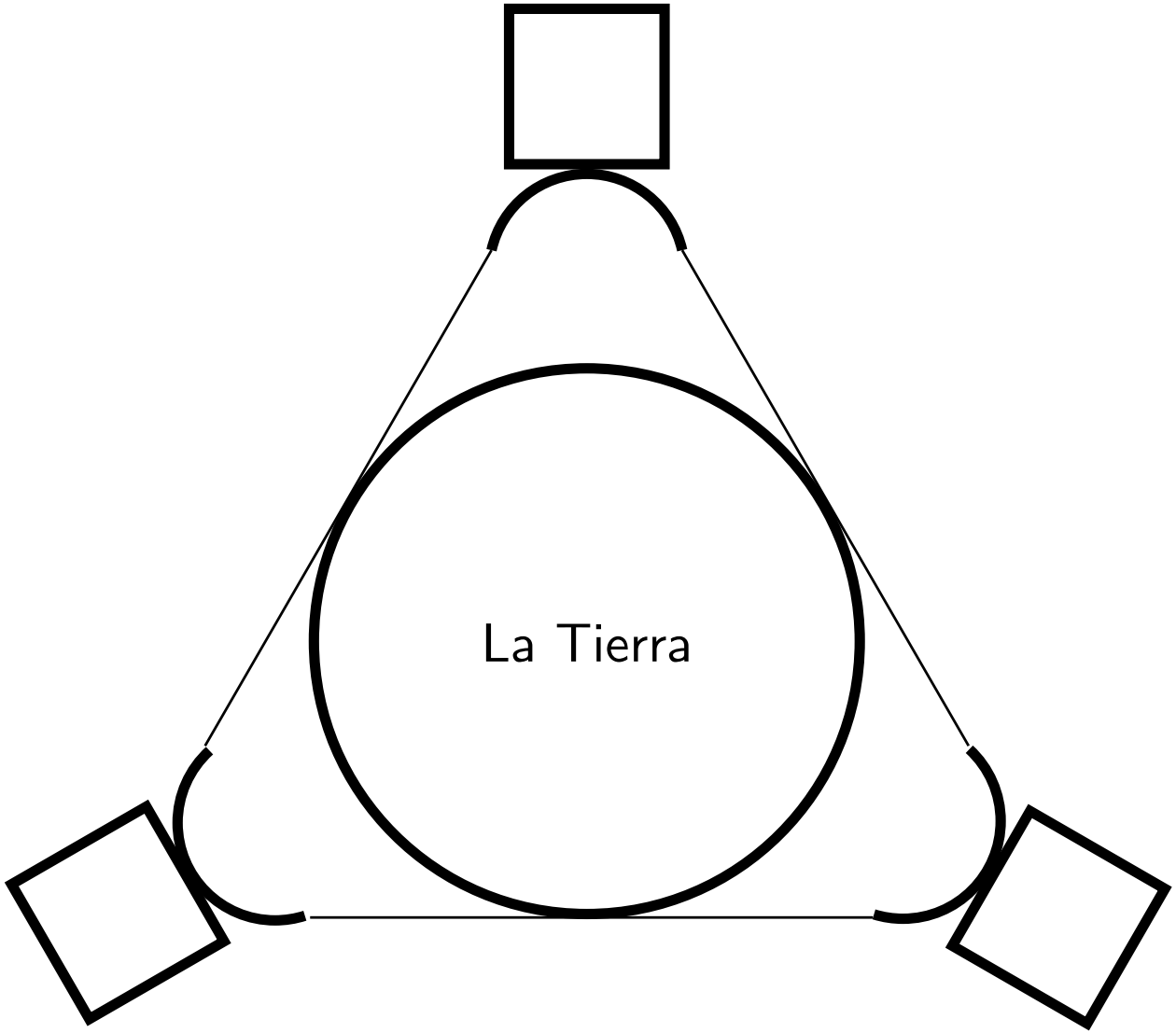


Figura H.1: Cobertura a nivel global utilizando tres satélites.

salir más barato utilizar satélites de comunicaciones geoestacionarios[3]. En un enlace de microondas por satélite, la estación emisora transmite los datos a una frecuencia diferente a la que el satélite las devuelve a La Tierra, para que ambas ondas no se interfieran. Dependiendo del grado de concentración de la antena utilizada para el canal descendente, el satélite puede transmitir una señal sobre un área geográfica más o menos amplia. Cuando la abertura es suficiente, con 3 satélites se asegura la cobertura en todo el planeta [8] (véase la Figura H.1).

La principal fuente de ruido en los enlaces de microondas son las interferencias provocadas por otros enlaces de microondas y por el mismo enlace. Las microondas se desvían y se reflejan con cierta facilidad en el terreno, en los edificios y en algunas capas de la atmósfera. Cuando las microondas que llegan hasta la antena receptora pertenecen a varias señales de datos, se habla de interferencias. Sin embargo, cuando lo que llega a la vez son la

[3]Un satélite geoestacionario permanece fijo en el cielo, lo que permite apuntar sobre él las antenas parabólicas y seleccionar a uno de entre una constelación de ellos. Todos los satélites geoestacionarios están a una distancia de 35.748 km de la superficie terrestre porque es el punto donde se anulan la fuerza gravitatoria y la fuerza centrífuga que actúan sobre el satélite [27, 32].

señal de datos y versiones retrasadas de esta, debido a reflexiones, se habla de desvanecimiento de trayectoria múltiple [32].

Apéndice I

Enlaces de rayos infrarrojos

Los rayos infrarrojos se comportan de una forma muy parecida a la luz: se reflejan con facilidad y no atraviesan los cuerpos opacos. La capacidad potencial de un enlace de infrarrojos es muy grande (véase la Figura F.1), pero no se pueden usar en exteriores a la luz del día porque el sol emite tanta

potencia en la banda visible como en la infrarroja [32]. Además, cualquier cuerpo caliente es también una fuente de rayos infrarrojos y constituye una fuente de ruido. Sin embargo, dentro de los edificios, habitaciones, etc., los enlaces de infrarrojos se usan desde hace tiempo para controlar el televisor, el vídeo, etc, y comienzan a ser una forma factible de evitar los cables del ratón, impresora, etc.[1]

[1] Los fabricantes de ordenadores llaman a esta tecnología Bluetooth [22].

Apéndice J

Enlaces de luz

La señalización óptica no guiada se ha usado durante milenios (piénsese por ejemplo en la comunicación de señales de humo o mediante espejos). Actualmente se utilizan lasers porque si no hay nada que lo impida, permiten transmitir una señal luminosa a una distancia enorme, ya que los rayos no

se dispersan en el espacio. Además, el ancho de banda de un enlace de luz es del mismo orden que el de la fibra óptica, lo que puede hacer que este tipo de comunicación sea ideal para transmisiones interplanetarias. Por desgracia, en la superficie terrestre los lasers son desviados con facilidad por las corrientes de convección provocadas cuando el aire se calienta. Por tanto, además de alinear perfectamente el emisor y el receptor, normalmente hay que instalar en el emisor algún sistema de lentes que disperse algo la luz y exista así un cierto margen de seguridad.

Apéndice K

Rayos X, rayos gamma y rayos cósmicos

Las posibilidades de transmisión de estas bandas de frecuencias son abrumadoras teniendo en cuenta los anchos de banda que disfrutan. Sin embargo

no se utilizan por dos motivos básicos:

1. **Son perjudiciales para la vida**.
   Estas radiaciones son capaces de romper los enlaces químicos que mantienen unidas las moléculas de las células. Son, por tanto, capaces de dañar el ADN, produciendo malformaciones en los fetos y cáncer en las personas adultas. A frecuencias más bajas (ondas de radio y microondas), la energía es absorbida por los tejidos vivos en forma de calor y de pequeñas corrientes de inducción.

2. **Son difíciles de manejar**.
   Los sistemas que manejen este tipo de radiaciones son muy complejos y costosos.

Apéndice L

La multiplexación de los enlaces

Existen numerosas ocasiones donde el medio de transmisión debe ser compartido por muchos emisores, como por ejemplo en las transmisiones a

través de señales de radio que se propagan por el vacío, aire, agua, etc. en todas direcciones. En estas situaciones donde el medio es único no queda más remedio que compartirlo mediante la multiplexación. En otros casos el medio es único simplemente por abaratar costes, ya que instalar un enlace de alta capacidad es más barato que instalar muchos de baja capacidad. Por ejemplo, es más barato multiplexar muchas señales de voz a través de una fibra óptica que instalar tantos pares trenzados como señales diferentes deseamos transmitir de forma simultánea [32].

Al dispositivo que permite multiplexar varias señales sobre un mismo medio de transmisión se le llama multiplexor y al que realiza el proceso inverso, desmultiplexor. Tras la desmultiplexación vuelven a ser recuperadas las señales originales. Un esquema simbólico de estos elementos se muestra en la Figura L.1.

Existen dos formas diferentes de multiplexación: en frecuencia y en tiempo, y ambas son equivalentes desde el punto de vista de su capacidad para transmitir datos. Sin embargo, al poseer diferentes características, dependiendo de la situación nos inclinaremos más por una que por la otra.

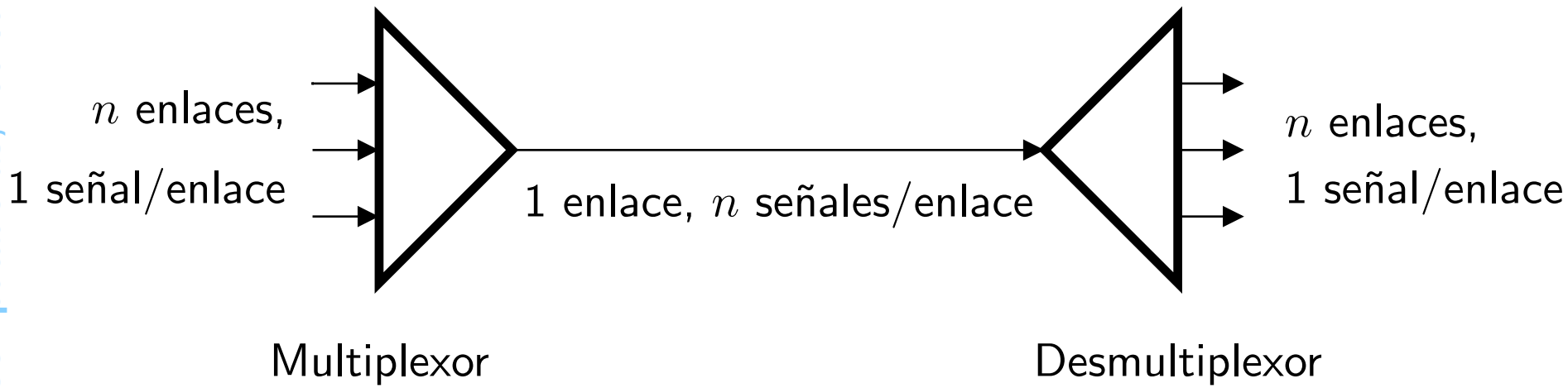


Figura L.1: Multiplexación/desmultiplexación de los enlaces físicos [27].

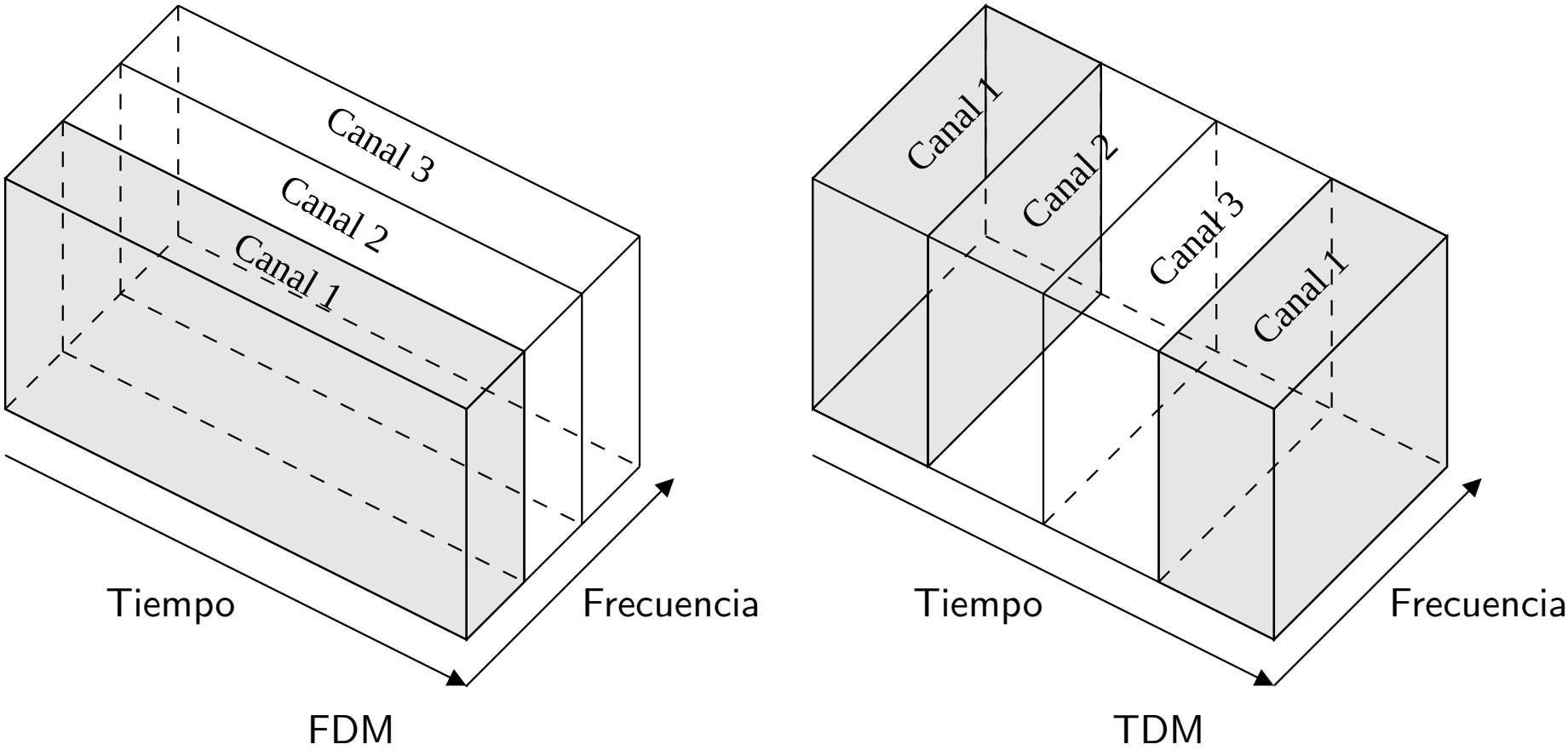


Figura L.2: FDM (multiplexación en frecuencia) versus TDM (multiplexación en el tiempo) [27]. Multiplexando en frecuencia los 3 canales se transmiten constantemente. Multiplexando en el tiempo los 3 canales se transmiten de forma cíclica, pudiendo en este caso utilizarse todo el ancho de banda del canal.

L.1. Multiplexación en el dominio de la frecuencia (FDM)

Cuando el ancho de banda W de un enlace es suficiente, N emisores pueden transmitir simultáneamente si cada uno de ellos utiliza una banda de frecuencias diferente. Cada emisor disfruta de forma continua de un ancho de banda igual a W/N. Por este motivo a la multiplexación en frecuencia se le llama también multiplexación por división de frecuencias o FDM (Frequency Division Multiplexing). A cada banda de frecuencias se le denomina **canal** (véase la Figura L.2).

La multiplexación en frecuencia se lleva a cabo desplazando adecuadamente los espectros de las señales a multiplexar mediante, por ejemplo, una modulación en amplitud, de forma que las bandas de frecuencias ocupadas sean diferentes. El multiplexor a continuación suma todas las señales. En la Figura L.3 se muestra un ejemplo donde se multiplexan en frecuencia tres señales.

La FDM es muy usada en telecomunicaciones analógicas y para llevarla a cabo se emplean, normalmente, las versiones para señales analógicas de

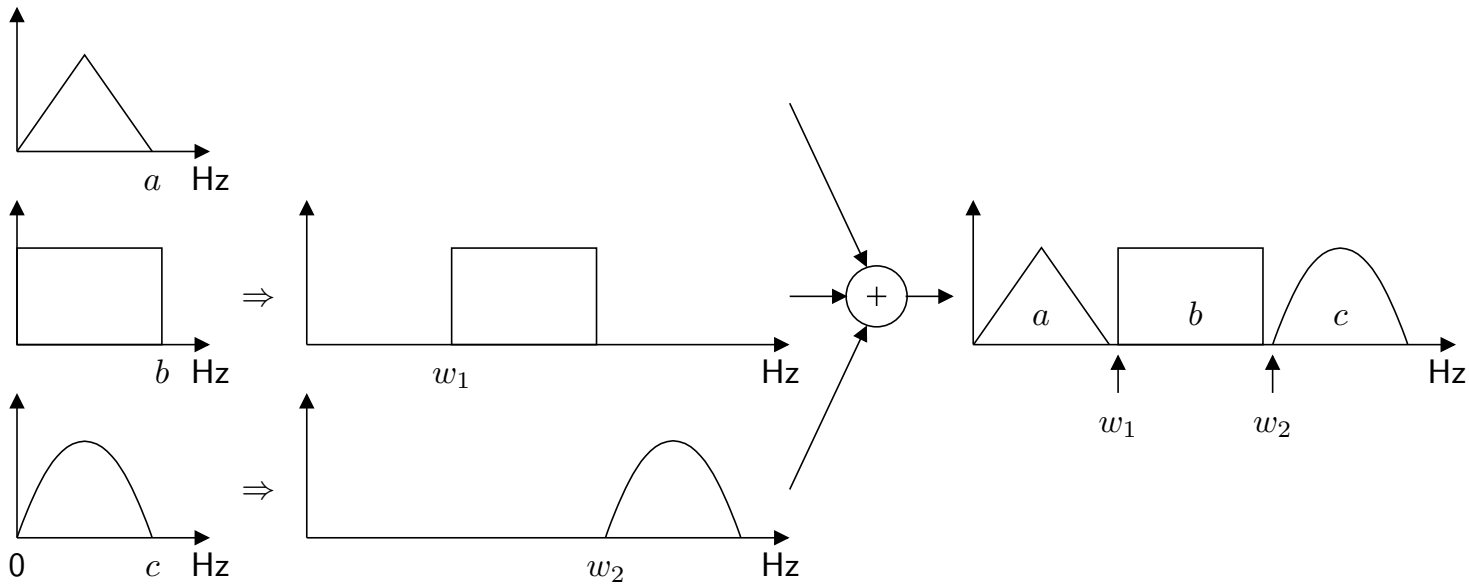


Figura L.3: Multiplexación en frecuencia de tres señales con anchos de banda a, b y c. w_1 y w_2 son las frecuencias de las portadoras usadas para desplazar los espectros de las señales con anchos de banda b y c usando modulación ($\Rightarrow$). Nótese que $w_1 > a$ y que $w_2 > a+b$ (la igualdad no se suele dar en la práctica porque siempre se dejan unas frecuencias de margen para evitar que las señales se solapen en el dominio de la frecuencia – aliasing –). Las tres señales pueden finalmente mezclarse (simplemente sumándose) para generar una señal multiplexada de tres canales.

ASK y FSK que se llaman AM (Amplitude Modulation) y FM (Frecuency Modulation) respectivamente. La diferencia entre AM y ASK, y FM y FSK es mínima: simplemente que las señales $s(t)$ son analógicas. Ejemplos típicos son la radio-difusión de TV (analógica), las señales de FM, etc., aunque también se aplica a señales digitales como ocurre en los modems full-duplex donde los emisores son también receptores (al mismo tiempo) y viceversa.

L.2. Multiplexación en el dominio del tiempo (TDM)

La idea de la multiplexación en tiempo o TDM (Time Division Multiplexing) consiste en que cada emisor puede hacer un uso exclusivo del enlace (disfrutando de todo el ancho de banda disponible) durante intervalos de tiempo suficientemente cortos. La duración de dichos intervalos depende de factores como: (1) el tiempo máximo de espera a la hora de emitir, (2) el número de emisores (que es en general variable), y (3) la prioridad de cada emisor (véase la Figura L.2). Además, los intervalos pueden ser todos iguales o variables. En este sentido, existen dos modalidades diferentes de TDM denominadas síncrona y asíncrona.

Aunque TDM se utiliza fundamentalmente para transmitir señales digitales, también puede usarse en el mundo analógico sin necesidad de digitalizar las señales. Esto se debe a que cualquier señal que esté limitada en banda puede ser representada sin pérdida de información por un número finito de muestras analógicas. En concreto, si f_m es la máxima componente en frecuencia, la señal queda especificada de forma única por sus valores a

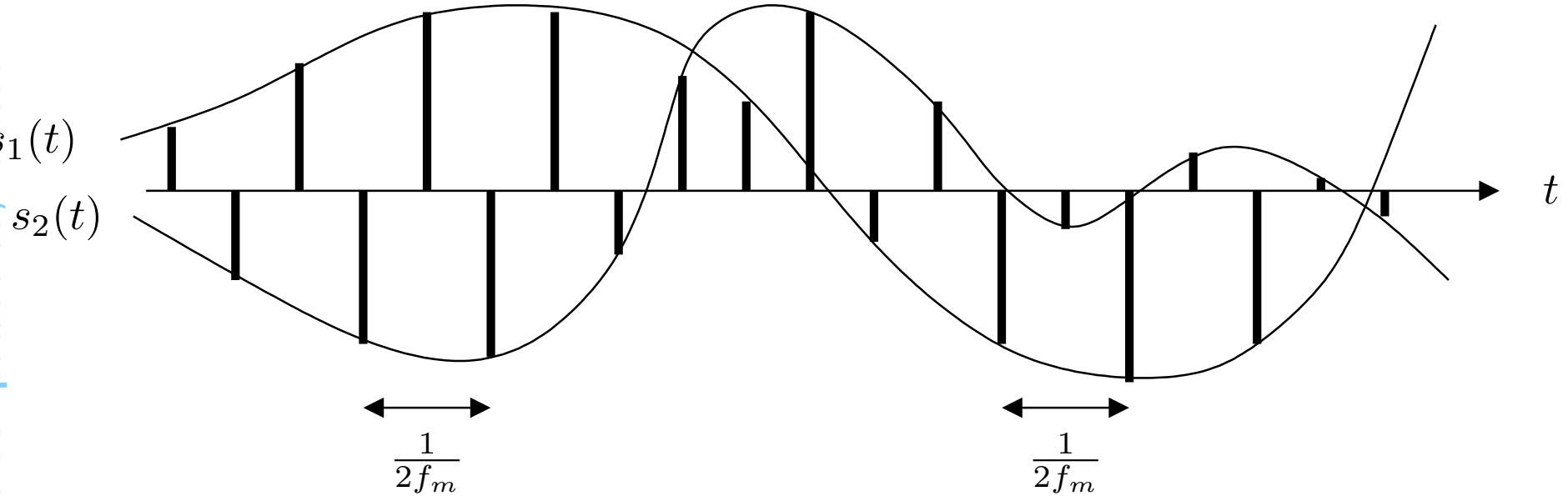


Figura L.4: Multiplexación en el tiempo de dos señales analógicas. f_m es la máxima componente de frecuencia de cada una de ellas.

intervalos de $\frac{1}{2f_m}$ segundos, según indica el **teorema del muestreo uniforme** [16]. Así, por ejemplo, si el ancho de banda del medio de transmisión es de $2f_m$ Hz, se pueden transmitir 2 señales con ancho de banda f_m intercalando las muestras de las 2 señales tal y como se muestra en la Figura L.4.

L.2.1. TDM síncrona

Cuando el multiplexor muestrea (asigna tiempo de transmisión) a todos los posibles emisores sin excepción, o lo que es lo mismo, cuando independientemente de que un emisor tenga o no que transmitir se le asigna un slot de tiempo, se habla de TDM síncrona. De esta forma, cada emisor potencial tiene garantizado su ancho de banda (periódicamente) y por lo tanto, esta modalidad de transmisión es equivalente a FDM con canales fijos. La TDM síncrona se utiliza especialmente en la transmisión de señales telefónicas, donde los interlocutores tienen reservado el canal, hablen o no hablen.

Por desgracia, esta forma de asignar los recursos del enlace puede llegar a ser bastante ineficiente cuando los emisores no tienen datos que transmitir. Lo que se hace para no perder la sincronía entre el multiplexor y el

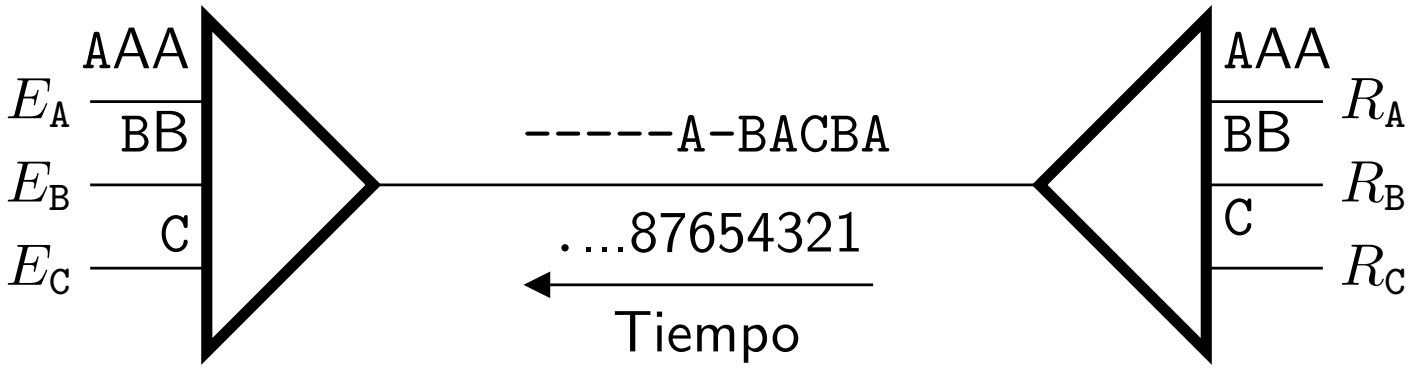


Figura L.5: Un ejemplo de multiplexación en el tiempo (TDM) síncrona (el tiempo corre de derecha a izquierda). E_A genera durante el intervalo de tiempo 1 una celda de bits A, otra durante el segundo intervalo y finalmente otra durante el tercer intervalo. E_B sólo genera dos celdas durante los dos primeros intervalos y E_C sólo genera una celda en el primer intervalo. Durante el sexto intervalo de tiempo no se transmite nada. Los receptores R_i reciben sus respectivas celdas porque el multiplexor y el desmultiplexor conocen la duración de cada celda.

desmultiplexor es enviar celdas de datos vacías. También puede ocurrir que el emisor termina de transmitir en medio de un intervalo de tiempo, con lo que sólo una parte de la celda estaría ocupada. En la Figura L.5 se muestra un ejemplo de transmisión donde se utiliza multiplexación en el tiempo síncrona.

Un ejemplo típico donde se utiliza TDM síncrona es en los sistemas de multiplexación SONET/SDH (Synchronous Optical NETwork/Synchronous Digital Hierarchy), que se emplea en los enlaces troncales (enlaces con una gran capacidad, en general fuertemente multiplexados) para la transmisión de conversaciones telefónicas en formato digital, donde cada canal telefónico posee una tasa de 64 kbps, 8.000 muestras/segundo, 8 bits/muestra y 4 kHz de ancho de banda).

L.2.2. TDM asíncrona

Esta forma de multiplexación temporal resuelve el problema del envío de celdas totalmente vacías existiendo en ese momento emisores que sí tienen datos que transmitir. La TDM asíncrona (también llamada **multiplexación temporal estadística** [27]) asigna un intervalo de tiempo (normalmente

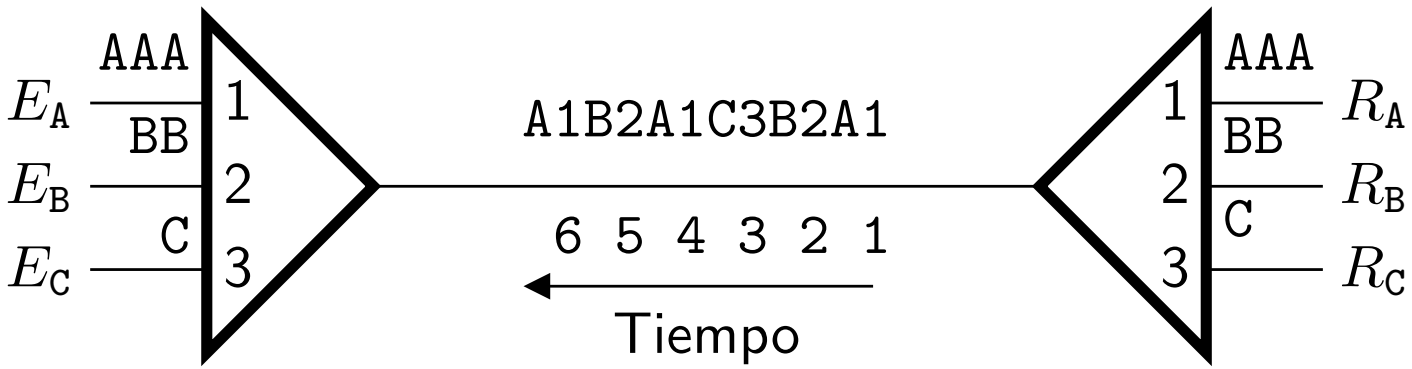


Figura L.6: Un ejemplo de multiplexación en el tiempo (TDM) asíncrona. La generación de datos es idéntica a la descrita en la Figura L.5. Sin embargo, ahora en cada celda viaja además una dirección que indica el origen o el destino de los datos de esa celda. Las direcciones son utilizadas por el desmultiplexor para saber la salida a la que enviar los datos de esa celda. Nótese que no existen celdas vacías.

fijo) cuando el emisor tiene algo que transmitir (véase la Figura L.6). El inconveniente de esta forma de multiplexación radica en que ahora debe transmitirse, junto con cada celda de datos, una cabecera de contiene la dirección del receptor destino de la celda. Esto supone un desperdicio de ancho de banda, al que llamaremos, en general, **overhead** del sistema de transmisión. De hecho, la TDM asíncrona sólo resulta eficiente si es poco probable que la mayoría de los emisores estén constantemente transmitiendo. Como veremos con posterioridad, en las redes de computadoras los emisores pasan largos periodos de tiempo sin transmitir. Por tanto, TDM asíncrona es la forma de multiplexación más utilizada en redes de computadoras.

Bibliografía

[1] How to connect to the MBone. http://www.live.com/mbone.

[2] GSM Association. GSM World. http://www.gsmworld.com.

[3] Joe Campbell. *Comunicaciones Serie. Guía de Referencia del Programador en C*. Anaya Multimedia, 1987.

[4] Douglas E. Comer. *Internetworking with TCP/IP. Principles, Protocols, and Architectures (4th Edition)*, volume 1. Prentice Hall, 2000.

[5] Defense Advanced Research Projects Agency (DARPA), http://www.rfc-editor.org/rfc/rfc793.txt. *RFC 793. The Transmission Control Protocol (TCP)*, 1981.

[6] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. *Hypertext Transfer Protocol – HTTP/1.1*. http://www.ietf.org/rfc/rfc2616.txt, June 1999.

[7] M. Ford, H.K. Lew, S. Spanier, and T. Stevenson. *Tecnologías de Interconectividad de Redes*. Prentice Hall, 1998.

[8] Behrouz Forouzan. *Introduction to Data Communications and Networking*. WCB/McGraw-Hill, 1998.

[9] Graham Glass. *Unix for Programmers and Users*. Prentice Hall, 1993.

[10] Fred Halsall. *Comunicaciones de Datos, Redes de Computadores y Sistemas Abiertos (4a Edición)*. Pearson Educación, 1998.

[11] Richard Wesley Hamming. Error Detecting and Error Correcting Codes. *The Bell System Technical Journal*, XXVI(2):147 – 160, April 1950. http://www.engelschall.com//~sb/hamming.

[12] Mark Handley. A MBone Scheduler. http://www-mice.cs.ucl.ac.uk/multimedia/projects/mice/mbone_review.html.

[13] IRIS-MBONE. Rediris - software mbone. http://www.rediris.es/mmedia/MboneSoft.es.html.

[14] James F. Kurose and Keith W. Ross. *Computer Networking: A Top-Down Approach Featuring the Internet (2nd Edition)*. Addison Wesley, 2003.

[15] James F. Kurose and Keith W. Ross. *Computer Networking: A Top-Down Approach Featuring the Internet (3rd Edition)*. Addison Wesley, 2005.

[16] Bhagwandas Pannalal Lathi. *Introducción a la Teoría y Sistemas de Comunicación*. Limusa Noriega Editores, 1994.

[17] Alberto León-García and Indra Widjaja. *Redes de Comunicación*. McGraw-Hill, 2002.

[18] Network Working Group, AT&T Research, http://www.rfc-editor.org/rfc/rfc793.txt. *Defending Against Sequence Number Attacks*, 1996.

[19] Alan V. Oppenheim, Alan S. Willsky, and S. Hamid Nawab. *Señales y Sistemas (2a edición)*. Prentice Hall, 1997.

[20] Soon J. Park. Mbone info. http://myhome.hanafos.com/~soonjp/mbone.html.

[21] Tom Perera. HISTORY, THEORY, & CONSTRUCTION OF THE ELECTRIC TELEGRAPH W1TP TELEGRAPH & SCIENTIFIC INSTRUMENT MUSEUMS. http://www.chss.montclair.edu/~pererat/pertel.htm, 2002.

[22] Larry L. Petterson and Bruce S. Davie. *Computer Networks: A System Approach (2nd Edition)*. Morgan Kaufmann, 2000.

[23] J. Postel. *RFC 768. The User Datagram Protocol (UDP)*. USC/Information Sciences Institute, http://www.rfc-editor.org/rfc/rfc768.txt, 1980.

[24] Gary R. Wright and W. Richard Stevens. *TCP/IP Illustrated*. Addison-Wesley, 1995.

[25] R. Sandberg, D. Goldberg, S. Kleiman, D. Walsh, and B. Lyon. Design and implementation of the sun network file system. In *Proceedings of the Summer 1985 Usenix Conference*, pages 119 – 131, June 1985. http://web.mit.edu/6.033/2002/wwwdocs/papers/nfs.pdf.

[26] K. Savetz, N. Randall, and Y. Lepage. MBONE: Multicasting Tomorrow's Internet. http://www.savetz.com/mbone.

[27] William Stallings. *Comunicaciones y Redes de Computadores (6a Edición)*. Prentice Hall, 2000.

[28] Sun Microsystems, Inc., http://www.rfc-editor.org/rfc/rfc1094.txt. *RFC 1094. NFS: Network File System Protocol Specification Version 2*, 1989.

[29] Sun Microsystems, Inc., http://www.rfc-editor.org/rfc/rfc1813.txt. *RFC 1813. NFS Version 3 Protocol Specification*, 1995.

[30] Sun Microsystems, Inc., http://www.rfc-editor.org/rfc/rfc1832.html. *RFC1832. XDR: External Data Representation Standard*, 1995.

[31] Sun Microsystems, Inc., http://www.rfc-editor.org/rfc/rfc3010.txt. *RFC 3010. Network File System (NFS) version 4 Protocol*, 2003.

[32] Andrew S. Tanenbaum. *Redes de Computadoras (3a Edición)*. Prentice Hall, 1997.

[33] Lloyd Wood. Lloyd's satellite constellations. http://www.ee.surrey.ac.uk/Personal/L.Wood/constellations.