3. LA CAPA DE ENLACE DE DATOS



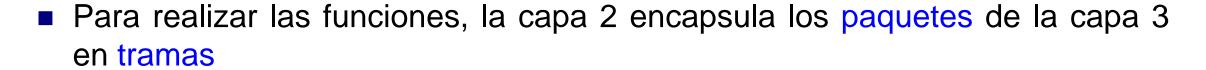
- 3.1 Cuestiones de diseño de la capa de enlace de datos
- 3.2 Detección y corrección de errores
- 3.3 Protocolos elementales de enlace de datos
- 3.4 Protocolos de ventana deslizante



- ☐ Tienen una **limitada** tasa de datos
- □ Retardan de propagación de los bits
- □ Se producen errores en la transmisión de los datos

3.1 Cuestiones de diseño de la capa de enlace de datos

- Las funciones de la capa 2 son dos:
 - 1. Maneja los errores de transmisión
 - 2. Regula el flujo de datos: emisores rápidos y receptores lentos



■ Trama:

- □ Encabezado (head)
- ☐ Carga útil: paquete (packet)
- □ Cola (tail)

Los controles de flujo y errores los realizan las capas 2 y 4

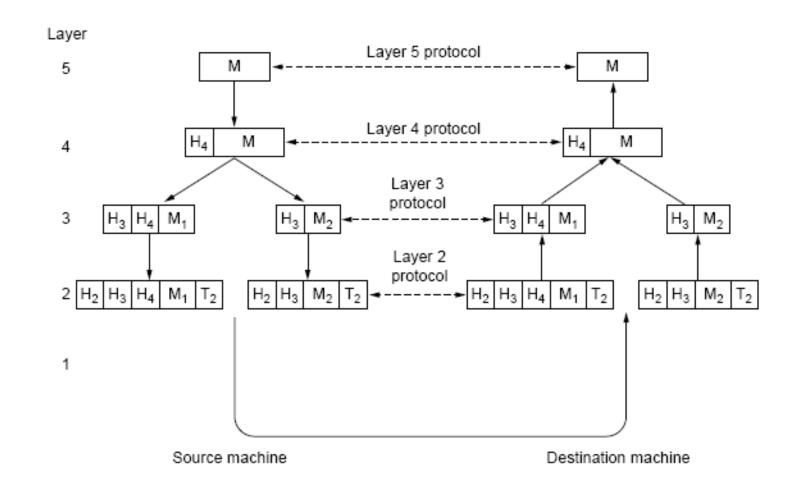


Fig. 1-15. Example information flow supporting virtual communication in layer 5.

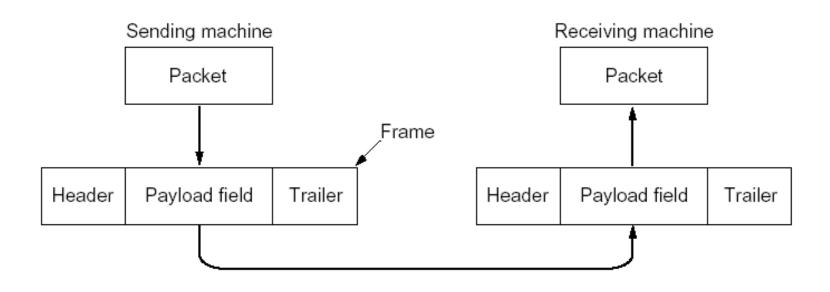


Fig. 3-1. Relationship between packets and frames.

3.1.1 Servicios a la capa de red

Servicio principal: transferir datos de la capa 3 en el origen a la capa 3 en el destino

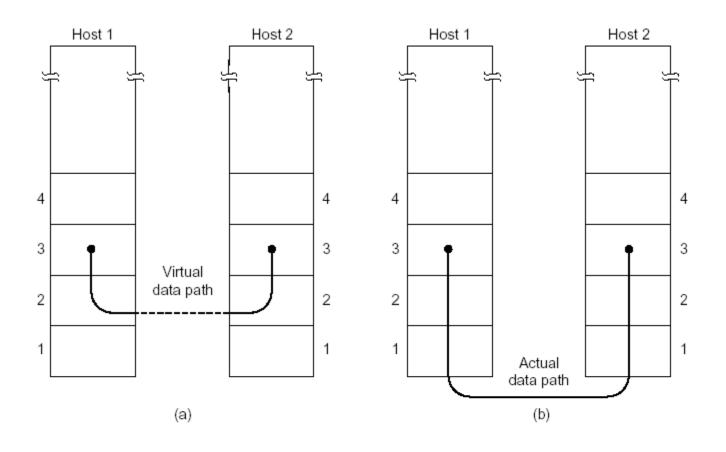


Fig. 3-2. (a) Virtual communication. (b) Actual communication.

Formas de dar servicio de la capa de enlace

- La capa 2 tiene tres maneras de dar el servicio a la capa 3
 - □ Sin conexión sin ACK.
 - □ Sin conexión con ACK.
 - Orientado a la conexión con ACK.

Servicio sin conexión sin ACK

- El origen envía tramas independientes al destino sin pedir confirmación (acknowledgment)
- No hay control de daño o pérdida de tramas debido al ruido en el medio
- En el destino, los datos se aceptan con o sin errores
- Servicio apropiado con medios de transmisión con baja tasa de errores
- Adecuado en:
 - □ redes LAN como Ethernet
 - □ aplicaciones de tiempo real: VoIP, videoconferencia



Servicio sin conexión con ACK

- El receptor avisa al emisor de qué manera llegó la trama
- Si la trama no ha llegado o llega dañada, el emisor lo reenvía
- ¿Cómo sabe el emisor que la trama no ha llegado a su destino?
- Usado en canales inestables: inalámbricos
- Con FO este servicio es innecesario, ¿por qué?
- Porque los datos llega y sin errores
- Utilizado en mensajería de texto de telefonía móvil (WhatsApp): se envía un mensaje y se recibe su confirmación

Servicio orientado a la conexión con ACK

- Es un servicio confiable
- Se simula una conexión numerando las tramas
- La numeración permite garantizar:
 - Recibir todas las tramas
 - En forma ordenada
 - □ Una sola vez
 - Sin errores
- Se simula una conexión porque las tramas llegan en orden y una sola vez
- El algoritmo podría implementarse en hardware

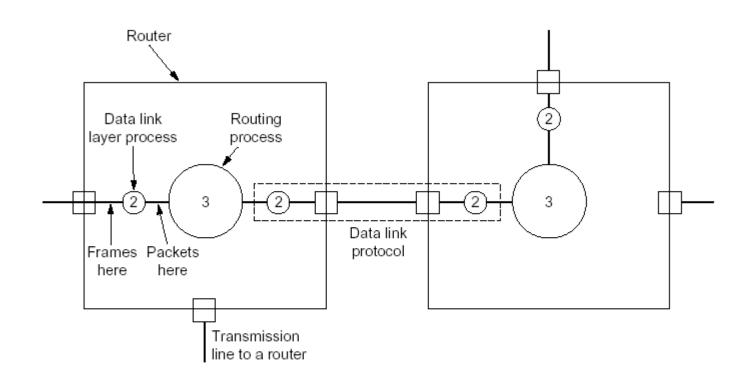


Fig. 3-3. Placement of the data link protocol.

3.1.2 Entramado

- Métodos para indicar el inicio y fin de una trama:
 - □ Inserción de intervalos de tiempo entre las tramas
 - □ Conteo de caracteres
 - □ Banderas de inicio y fin, con relleno de caracteres
 - □ Banderas de inicio y fin, con relleno de bits
 - □ Violación de codificación de la capa física

Inserción de intervalos de tiempo

- Pausas entre tramas
 - ☐ Similar a los espacios de tiempo entre palabras
- Riesgo:
 - Los intervalos podrían ser eliminados
 - □ Por error podrían insertarse intervalos dentro de una trama



 Longitud de la trama: campo en la cabecera para indicar el número de caracteres de la trama

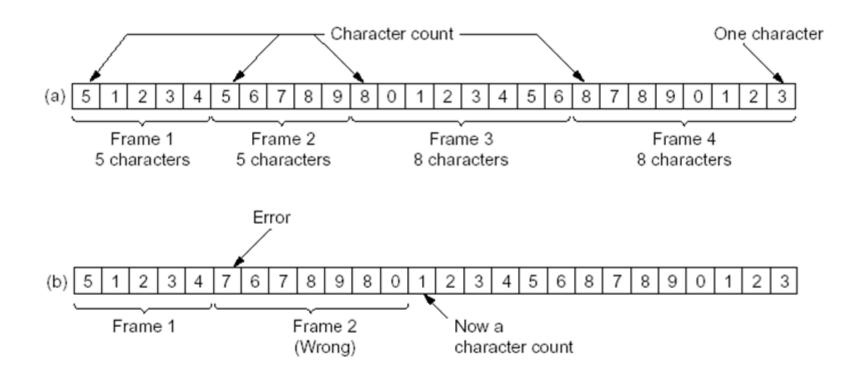
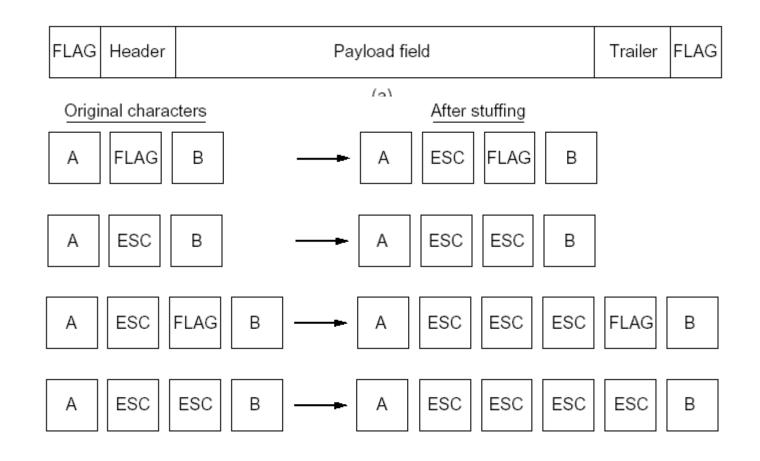


Fig. 3-4. A character stream. (a) Without errors. (b) With one error.

.

Banderas con relleno de caracteres

- Inicio y fin de una trama con un byte especial
- **•** 01111110
- La bandera sincroniza al receptor con el emisor
- Si el receptor pierde sincronía, puede esperar la siguiente bandera
- Dos banderas consecutivas señalan el fin de una trama e inicio de la siguiente
- Podría ocurrir que el patrón de bits de la bandera aparezca entre los datos
- Para solucionar se usa el byte ESC

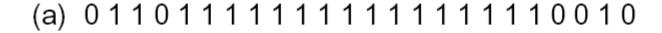




Banderas con relleno de bits

- Cada trama inicia y termina con 011111110 (seis 1s)
- Si la capa 2 encuentra cinco 1s consecutivos en los datos, inserta un 0
- Si el receptor ve cinco 1s de entrada seguido de un 0, extrae el 0 de relleno

Bits de relleno en los datos





(c) 01101111111111111110010

Violaciones de codificación de la capa física

- Algunas LANs representan los bits de datos usando codificación Manchester
 - ☐ Alto bajo: bit 1
 - ☐ Bajo alto: bit 0
- alto-alto y bajo-bajo no se usa para datos
- Algunos protocolos usan alto-alto y bajo-bajo para delimitar tramas

3.1.3 Formas de controlar los errores

- Primero es necesario implementar un sistema de detección los errores
- Luego se retroalimenta al emisor con tramas de control ACK NACK
- Son tramas de confirmación positiva o negativa
- ACK Acknowledgement
- ACK: envíe la siguiente trama
- NACK: Retransmita la trama dañada
- Si la trama de datos no llega al receptor, no hay retroalimentación:
 - □ Inicio de temporizador en el emisor
 - □ Expira el temporizador retransmite
 - ☐ Tramas numeradas para no duplicar si el receptor si recibió la trama pero ACK se perdió

3.1.4 Control de flujo

- Si existen emisores rápidos y/o receptores lentos
- Hay saturación, sobrecarga o desbordamiento del receptor
- Para evitarlo se controla el flujo de tramas del emisor hacia el receptor:
 - □ Basado en retroalimentación al emisor. Se usa en la Capa de Enlace de Datos
 - Basado en tasa. El protocolo limita la tasa a la que el emisor puede transmitir los datos.
 Se asigna un determinado ancho de banda. Usado en la Capa de Red

3.2 Detección y corrección de errores



- □ Última milla, si es de cobre (*local loop,* bucle local, bucle de abonado, o planta externa del proveedor del servicio)
- Comunicación inalámbrica
- Debido a ráfagas de ruido, los errores aparecen en ráfagas de bits, no individualmente
- Desventaja de errores en ráfaga: más difíciles de detectar y corregir
- Ventaja: dañan una o máximo dos tramas

3.2.1 Códigos de corrección de errores

- Hay dos estrategias:
- Códigos de corrección de errores
 - □ Incluir información redundante en la trama para que el receptor corrija el error
- Códigos de detección de errores
 - □ Incluir suficiente información para que el receptor sepa que ha ocurrido un error



- ☐ Es suficiente usar códigos de detección de errores
- Wireless: tasa alta de errores debido a ruido ambiental, descargas eléctricas atmosféricas, interferencia con otras señales
 - □ Es mejor usar códigos de corrección de errores

En dos palabras codificadas, el número de bits en los que difieren es h

10001001

10110001

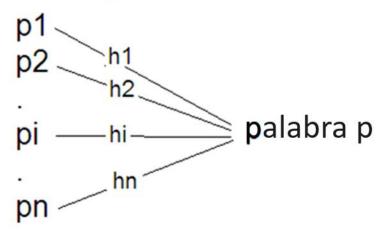
XOR 00111000

- En el ejemplo, h = 3
- h es la distancia de Hamming



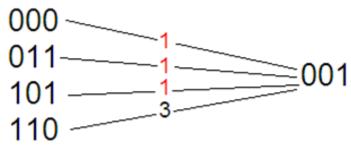
- Se puede construir una lista completa de palabras válidas
- Las palabras deben ser lo suficientemente distintas entre sí
- Para ello, h entre dos palabras no será menor a una distancia mínima m
- La facilidad para detectar y corregir errores depende de h
- Si una palabra tiene un h < m con alguna de las palabras legales, se detectará el error
- Si h < m, esta cantidad h de bits no es posible convertir una palabra válida en otra palabra válida

Lista legal



Si hi < m, se detecta error

Lista legal m = 2



1 < 2, se detecta error

Corrección de errores, 1era, forma

- Para corregir e bits errados o menos se necesita un código de distancia m =
 2e + 1
- Las palabras legales están tan separadas que alterando hasta e bits todavía esta más cerca de la palabra original

Ejemplo de corrección de errores

En vez de: 00
 Se tiene: 0000000000
 0000011111
 10
 1111111111
 1111111111

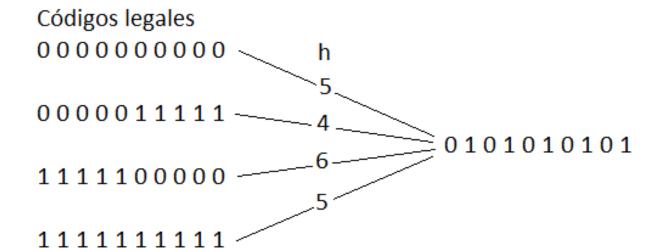
$$\mathbf{m} = 5$$

$$m = 2e + 1$$

$$e = (m - 1)/2 = (5 - 1)/2 = 2$$

Se pretende corregir hasta e = 2 errores.

$$m = 2e + 1$$
$$m = 5$$



Las distancias h son mayores a e = 2

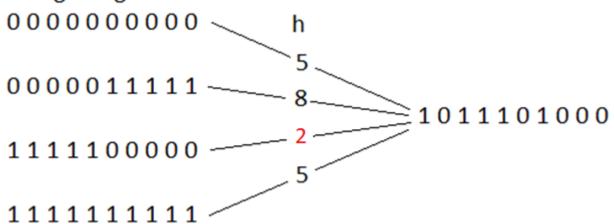
No se pueden corregir los errores

Se pretende corregir hasta e = 2 errores.

$$m = 2e + 1$$

$$m = 5$$

Códigos legales



Corrección de errores. Códigos de Hamming. 2da. forma

- Se desea corregir errores individuales
- Se debe determinar el número de bits redundantes r
- El mensajes tiene *m* bits
- n longitud de la trama
- = n = m + r

¿Cuál es el número mínimo de bits redundantes r?

- m = longitud del mensaje
- Hay 2^m mensajes legales
- Si m = 2 hay 4 mensajes legales: 00, 01, 10, 11
- Haremos que por cada mensaje legal haya n palabras ilegales a una distancia 1 del mensaje
- Entonces, el total de mensajes ilegales tiene que ser $n2^m$
- Palabras legales + ilegales: 2^m + n2^m = (n + 1)2^m
- Debe hacerse cumplir que: legales + ilegales ≤ 2ⁿ
- (n + 1) 2^m ≤ 2ⁿ
- n = m + r; $(m + r + 1) 2^m \le 2^{m+r}$
- $m + r + 1 \le 2^r$
- Por tanteo, si m = 2 entonces r = 3

41

Ejemplo

```
m = 2; r = 3; n = m + r = 5
legales 2^m = 4 ilegales n2^m = 20 con errores de 1 bit

00000 10000 01000 00010 00001

01011 11011 00011 01111 01001 01010

00100 11100 10000 10110 10101

01111 10111 11011 11101 11110
```

- Legales + ilegales = $2^m + n \ 2^m = (n + 1)^* \ 2^m = 24$
- Legales + ilegales <= 2ⁿ
- **2**4 <= 32

Ejemplo de Códigos de Hamming

```
bits m = 7
                                       m + r + 1 \le 2^r; r = 4
                         Posición k
       1 2 3 4 5 6 7 8 9
ASCII
 2<sup>n</sup>
```

$$3 = 1 + 2$$

$$5 = 1 + 4$$

$$6 = 4 + 2$$

```
Posición k
      1 2 3 4 5 6 7 8 9 10 11
ASCII
2<sup>n</sup>
```

Posición k 6 **ASCII** 2ⁿ Ing. Raúl Ortiz Gaona

47

Paridad par de 1s

		1	2	3	4	Posi 5	ción k <mark>6</mark>	7	8	9	10	11
A	ASCII			1		0	0	1		0	0	0
	2 ⁿ											
	1	0		1		0		1		0		0
_	2			1			0	1			0	0
	4					0	0	1				
Ing. Raúl Ortiz Gao	8 ona									0	0	0

Paridad par de 1s

		1	2	3	4	Posi 5	ción k	7	8	9	10	11
	ASCII			1		0	0	1		0	0	0
	2 ⁿ											
	1	0		1		0		1		0		0
	2		0	1			0	1			0	0
	4					0	0	1				
Ing. Raúl Ortiz G	8 aona									0	0	0

Paridad par de 1s

		1	2	3	4	Posi	ición l	7	8	9	10	11
	ASCII			1		0	0	1		0	0	0
	2 ⁿ											
	1	0		1		0		1		0		0
	2		0	1			0	1			0	0
	4				1	0	0	1				
Ing. Raúl Ortiz G	8 aona								0	0	0	0

Código de Hamming

1	2	3	4	5	ición l	7	8	9	10	11
0	0	1	1	0	0	1	0	0	0	0

2 ⁿ												
1	0		1		0		1		0		0	
2		0	1			0	1			0	0	
4				1	0	0	1					
Q								0	Λ	0	Λ	

Simulación de error en los datos:

Dato enviado: 1001000

Dato recibido con error: 1001001

1	2	3	4	5	1cion i	7	8	9	10	11
1	1	1	1	0	0	1	1	0	0	1

D - - ! - ! / - . | . |

2 ⁿ											
1	1		1		0		1		0		1
2		1	1			0	1			0	1
4				1	0	0	1				
								_	_		_

Posición de los bits de paridad: 1 2 4 8
Bits recibidos: 0 0 1 0
Bits calculados en el receptor: 1 1 1 1
error error

Sumar las potencias de 2 en donde hubo error:

$$1 + 2 + 8 = 11$$

El bit 11 está errado. Hay que invertir su valor

Error: 1001001

Dato corregido: 1001000



- Los Códigos de Hamming sólo corrigen errores individuales
- Hay un truco para corregir códigos de ráfaga
- k palabras se disponen en matriz
- No se transmite por filas, sino por columnas
- Un error de ráfaga de longitud k, dañará a lo mucho 1 bit de cada una de las k palabras



Char.	ASCII	Check bits
Н	1001000	12 4 8 I 00110010000
a	1100001	10111001000
m	1101101	11101010101
m	1101101	11101010101
i	1101001	01101011001
n	1101110	01101010110
g	1100111	01111001111
	0100000	10011000000
С	1100011	11111000011
0	1101111	10101011111
d	1100100	11111001100
е	1100101	00111000101
		Order of bit transmission

3.2.2 Códigos de detección de errores

- Bit de paridad
 - Paridad par: cantidad par de 1s
 - Paridad impar: cantidad impar de 1s
- Códigos con bit de paridad tienen distancia mínima m = 2

10100011

11100010

XOR 01000001 h = 2

- Cualquier error de un solo bit será detectado
- Es un sistema para detectar errores individuales

Código de redundancia cíclica CRC

- Una cadena de bits se trata como un polinomio, con coeficientes 0 y 1
- Una trama de k bits se considera como un polinomio de k términos de grado k - 1
- $\mathbf{b}_{k-1} \mathbf{x}^{k-1} + \mathbf{b}_{k-2} \mathbf{x}^{k-2} + \dots + \mathbf{b}_1 \mathbf{x} + \mathbf{b}_0$
- \blacksquare 101011 = 1x⁵ + 0x⁴ + 1x³ + 0x² + 1x + 1

57

■ La suma y la resta se reducen a un XOR

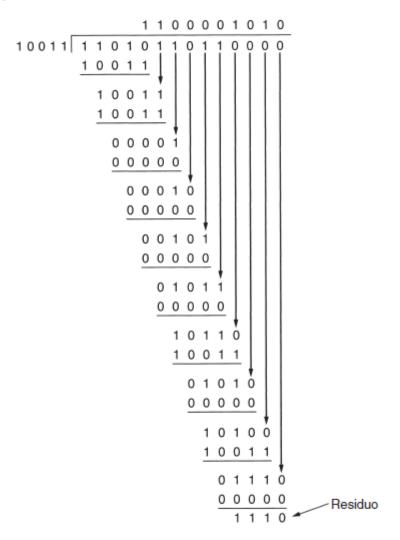
10011011 ± 11001010 01010001

- Emisor y receptor acuerdan el generador G(x)
- El mensaje M(x) de m bits, debe ser más grande que G(x)
- Se añade la suma de verificación CRC al final de M(x)
- T(x) = M(x) + CRC
- T(x) trama que se transmite
- T(x) debe ser divisible para G(x)
- Si hay residuo, hay un error de transmisión

M(X) Trama : 1101011011

G(X) Generador: 10011

Mensaje tras anexar 4 bits cero: 1 1 0 1 0 1 1 0 1 1 0 0 0 0



T(X) Trama transmitida: 1 1 0 1 0 1 1 0 1 1 1 1 1 1 0

3.3 Protocolos elementales de enlace de datos

Archivo protocol.h

```
#define MAX PKT 1024
                                             /* determina el tamaño del paquete
                                                en bytes */
typedef enum {false, true} boolean;
                                            /* tipo booleano */
                                            /* números de secuencia o
typedef unsigned int seg nr:
                                                confirmación */
typedef struct {unsigned char data[MAX_PKT];} packet;/* definición de paquete */
typedef enum {data, ack, nak} frame kind; /* definición de frame kind */
typedef struct {
                                             /* las tramas se transportan en
                                                esta capa */
                                             /* ¿qué clase de trama es? */
 frame kind kind;
                                             /* número de secuencia */
 seq nr seq;
                                             /* número de confirmación de
 seq nr ack;
                                                recepción */
                                             /* paquete de la capa de red */
 packet info;
} frame;
/* Espera que ocurra un evento; devuelve el tipo en la variable event. */
void wait for event(event type *event);
/* Obtiene un paquete de la capa de red para transmitirlo por el canal. */
void from network layer(packet *p);
/* Entrega información de una trama entrante a la capa de red. */
void to network layer(packet *p);
/* Obtiene una trama entrante de la capa física y la copia en r. */
void from physical layer(frame *r);
```

```
/* Pasa la trama a la capa física para transmitirla. */
void to physical layer(frame *s);
/* Arranca el reloj y habilita el evento de expiración de temporizador. */
void start timer(seq nr k);
/* Detiene el reloj e inhabilita el evento de expiración de temporizador. */
void stop timer(seg nr k);
/* Inicia un temporizador auxiliar y habilita el evento ack timeout. */
void start ack timer(void);
/* Detiene el temporizador auxiliar e inhabilita el evento ack timeout. */
void stop ack timer(void);
/* Permite que la capa de red cause un evento network_layer_ready. */
void enable network layer(void);
/* Evita que la capa de red cause un evento network layer ready. */
void disable network layer(void);
/* La macro inc se expande en línea: incrementa circularmente k. */
#define inc(k) if (k < MAX_SEQ) k = k + 1; else k = 0
```



- □ Protocolo símplex sin restricciones
- □ Protocolo símplex de parada y espera
- ☐ Protocolo simplex para un canal ruidoso

3.3.1 Protocolo símplex sin restricciones

- Es sencillo
- Es irreal
- Los datos se transmiten sólo en una dirección
- Las capas de red del emisor y receptor siempre están listas
- Se ignora el tiempo de procesamiento
- Hay un espacio infinito de buffer
- Canal no produce errores ni pierde tramas

```
typedef enum {frame_arrival} event_type;
#include "protocol.h"
void sender1(void)
 frame s;
                               /* búfer para una trama de salida */
                                /* búfer para un paquete de salida */
  packet buffer;
  while (true) {
     from_network_layer(&buffer); /* consigue algo que enviar */
     s.info = buffer; /* lo copia en s para transmisión */
                               /* lo envía a su destino */
     to_physical_layer(&s);
                                 /* Mañana, y mañana, y mañana,
                                   Se arrastra a este mísero paso de día
                                     a día
                                   Hasta la última sílaba del tiempo
                                     recordado
                                       -Macbeth, V, v */
void receiver1(void)
 frame r;
                                /* ocupado por wait, pero no se usa aquí */
  event type event;
  while (true) {
     wait_for_event(&event); /* la única posibilidad es frame_arrival */
     from_physical_layer(&r); /* obtiene la trama entrante */
     to network layer(&r.info); /* pasa los datos a la capa de red */
```



Único evento posible: llega tramas sin daños

3.3.2 Protocolo simplex de parada y espera

- El receptor requiere un tiempo para recibir y procesar la trama que llega
- Se evita que el emisor sature al receptor
- Se supone que el canal de comunicaciones está libre de errores
- El tráfico de datos es símplex
- Solución: receptor retroalimenta al emisor
- Luego de enviar una trama, el emisor espera que llegue una trama ACK
- La trama ACK autoriza al emisor para que envíe la siguiente trama
- Este protocolo se llama de «parada y espera»

```
void sender2(void)
 frame s;
                                /* búfer para una trama de salida */
                               /* búfer para un paquete de salida */
  packet buffer;
                                /* frame arrival es la única posibilidad */
  event type event;
  while (true) {
     from network layer(&buffer); /* consigue algo que enviar */
     s.info = buffer; /* lo copia en s para transmisión */
    to_physical_layer(&s); /* adiós a la pequeña trama */
     wait for event(&event); /* no procede hasta que recibe la señal de
                                    continuación */
void receiver2(void)
                              /* búferes para las tramas */
 frame r, s;
 event_type event;
                                /* frame arrival es la única posibilidad */
  while (true) {
     wait for event(&event); /* la única posibilidad es frame arrival */
    from_physical_layer(&r); /* obtiene la trama entrante */
     to network layer(&r.info); /* pasa los datos a la capa de red */
     to physical layer(&s);
                                /* envía una trama ficticia para informar
                                    al emisor */
```

■ En realidad hay flujo de información en ambas direcciones en una alternancia estricta

3.3.3 Protocolo símplex para un canal ruidoso

- El canal comete errores o pierde tramas
- Puede ser que el ACK enviado por el receptor se pierda
 - □ El emisor lo vuelve a enviar duplicándose la trama
 - ☐ Para evitar problemas se numera las tramas

- El número de secuencia es de 1 bit (0 ó 1)
- Éste se incrementa módulo 2
- Se transmite también en una sola dirección
- Los protocolos en los que el emisor espera confirmación se llaman ARQ (Automatic Repeat reQuest)

```
/* debe ser 1 para el protocolo 3 */
#define MAX SEQ 1
typedef enum {frame arrival, cksum err, timeout} event type;
#include "protocol.h"
void sender3(void)
  seq nr next frame to send;
                                            /* número de secuencia de la siguiente
                                                trama de salida */
  frame s;
                                            /* variable de trabajo */
                                            /* búfer para un paquete de salida */
  packet buffer:
  event type event;
  next frame to send = 0;
                                            /* inicializa números de secuencia de
                                                salida */
  from network layer(&buffer);
                                            /* obtiene el primer paquete */
  while (true){
     s.info = buffer;
                                          /* construye una trama para transmisión */
                                            /* inserta un número de secuencia en la
     s.seq = next frame to send;
                                                trama */
                                            /* la envía a su destino */
     to physical layer(&s);
     start timer(s.seq);
                                            /* si la respuesta tarda mucho, expira el
                                                temporizador */
     wait_for_event(&event);
                                            /* frame arrival, cksum err, timeout */
     if (event == frame arrival){
           from physical layer(&s);
                                             /* obtiene la confirmación de recepción */
            if (s.ack == next frame to send){
                  stop timer(s.ack); /* desactiva el temporizador */
                  from network layer(&buffer);
                                                   /* obtiene siguiente a enviar */
                  inc(next_frame_to_send); /* invierte next_frame_to_send */
```

```
void receiver3(void)
  seq nr frame expected;
  frame r, s;
  event type event;
  frame expected = 0;
  while (true){
     wait_for_event(&event);
                                             /* posibilidades: frame_arrival, cksum_err */
     if (event == frame_arrival){
                                             /* ha llegado una trama válida. */
                                              /* obtiene la trama recién llegada */
           from_physical_layer(&r);
           if (r.seq == frame_expected) { /* esto es lo que hemos estado esperando. */
                  to_network_layer(&r.info); /* pasa los datos a la capa de red */
                  inc(frame_expected);
                                             /* para la próxima se espera el otro número
                                                 de secuencia */
                                             /* indica la trama cuya recepción se está
           s.ack = 1 - frame_expected;
                                                 confirmando */
          to_physical_layer(&s);
                                             /* envía confirmación de recepción */
```

3.4 Protocolos de ventana deslizante



- Otra forma es usar un circuito para datos en ambas direcciones
- El campo *kind* de la cabecera de la trama indica si es de datos o ACK

Técnica de superposición o piggybacking

- Se anexa un ACK a una trama de datos de retorno, usando el campo ack del encabezado
- ACK viaja gratis
- Se aprovecha mejor el BW del canal
- Hay menos interrupciones de "ha llegado una trama"

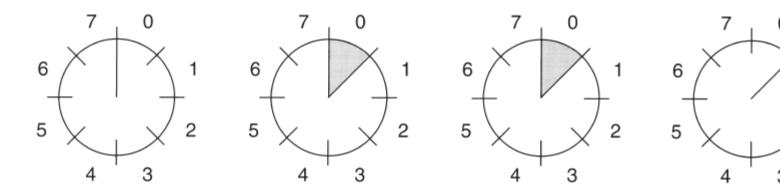
- Problema: Podría expirar el temporizador del emisor y retransmitir la trama porque el receptor no tiene datos que enviar
- Solución: El receptor espera un tiempo. Si el receptor no tiene datos que enviar, envía solo un ACK

Tres protocolos de ventana deslizante

- Son bidireccionales:
 - □ Protocolo de ventana corrediza de un bit
 - Protocolo de retroceso N
 - □ Protocolo de repetición selectiva
- Cada trama tiene un número de secuencia
- El emisor registra las tramas enviadas aun no confirmadas
- El receptor tiene una ventana de tramas a aceptar

- Estos protocolos entregan en orden los paquetes a la capa 3
- En el emisor:
 - □ Si llega un paquete nuevo de la capa 3, el extremo superior de la ventana avanza 1
 - □ Al llegar un ACK, el extremo inferior de la ventana avanza 1







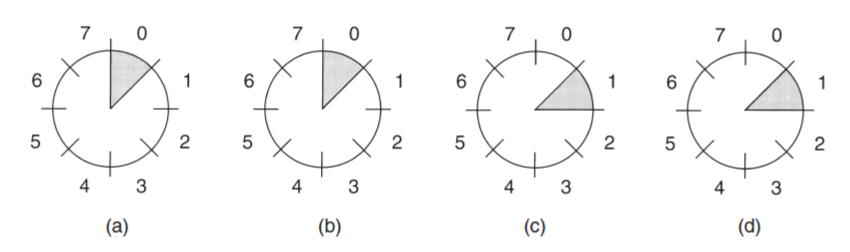


Figura 3-13. Ventana corrediza de tamaño 1, con un número de secuencia de 3 bits. (a) Al inicio. (b) Tras la transmisión de la primera trama. (c) Tras la recepción de la primera trama. (d) Tras recibir la primera confirmación de recepción.

3.4.1 Protocolo de ventana corrediza de 1 bit

- Utiliza parada y espera
- Las tramas se numeran con 0, 1, 0, 1, ...
- Sólo una de las máquinas puede iniciar la transmisión
- Si las 2 máquinas inician la transmisión al mismo tiempo hay funcionamiento anormal

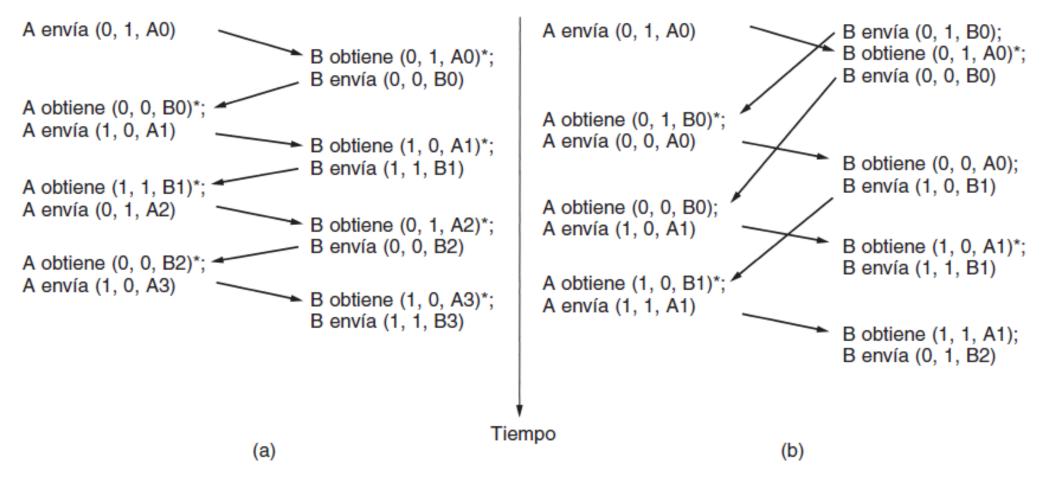


Figura 3-15. Dos escenarios para el protocolo 4. (a) Caso normal. (b) Caso anormal. La notación es (secuencia, confirmación de recepción, número de paquete). Un asterisco indica el lugar en que una capa de red acepta un paquete.

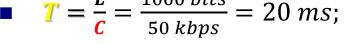
3.4.2 Protocolo que usa retroceso N: Canalización

- Enlace satelital de C = 50 kbps
- Satélite geoestacionario 36.000 km
- Ida y vuelta al emisor es 4 x 36.000 km
- Retardo de propagación de ida y vuelta P

$$P = 4 \times \frac{36.000 \, km}{300.000 \, km/s} = 480 \, ms$$

- Longitud de la trama L = 1000 bits
- Tiempo de transmisión de la trama T

$$T = \frac{L}{C} = \frac{1000 \ bits}{50 \ kbps} = 20 \ ms;$$



- Tiempo de transmisión y recibir ACK de una trama TP = T + P = 20 ms + 480 ms = 500 ms
- El emisor espera la confirmación de la trama enviada antes de enviar las siguiente trama
- Con ACK corto, el uso del canal $U1 = T/(TP)x100 = (20s/500 ms) \times 100 = 4.0\%$





Solución

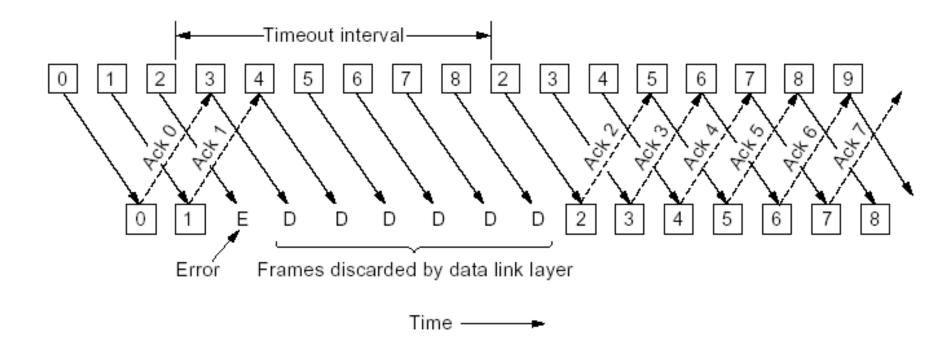
- **Canalización**. El emisor envía w tramas durante el tiempo total TP = 7 + P = 500 ms
- ¿Cuántas tramas puede enviar el emisor, o cuál es el tamaño de la ventana w?
- w = TP / T = 500ms / 20ms = 25 tramas
- El emisor ocupa el canal por 500 ms para enviar 25 tramas
- El emisor espera los ACKs de todas ellas antes de enviar las siguientes 25 tramas
- Tiempo para transmitir y recibir ACKs de 25 tramas TT
- $TT = TP + 24T = 500 \, ms + 24x20 \, ms = 980 \, ms$
- 247 es el tiempo que demora en llegar al receptor los ACKs de las tramas restantes
- Uso del canal U2 = tiempo de transmisión de w tramas TP/TT
- $U2 = (500 \, ms \, / \, 980 \, ms) x 100 = 51\%$
- Mucho mayor uso del canal

- Al terminar de enviar la trama 26 en t = 520 ms, llega la confirmación de recepción de la trama 1
- Entonces las confirmaciones llegan cada 20 ms

Métodos para manejar errores en canalización

- Retroceso n
- 2. Repetición selectiva

Retroceso n



Retroceso n continuación

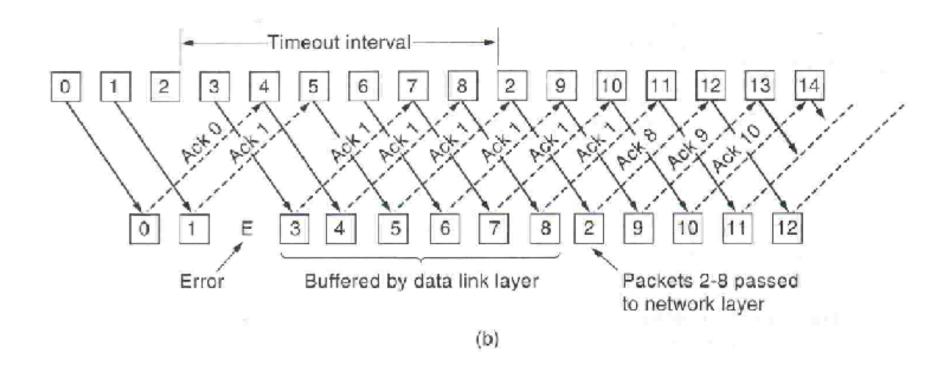
- Tamaño de la ventana del receptor = 1
- El receptor descarta las tramas subsiguientes
- El canal de comunicación no se aprovecha si la tasa de errores es alta



Repetición selectiva

- Ventana del receptor > 1
- Se descarta la trama dañada recibida
- Siguientes tramas recibidas correctamente se almacenan en buffer
- Emisor transmite sólo la última trama sin ACK

Repetición selectiva



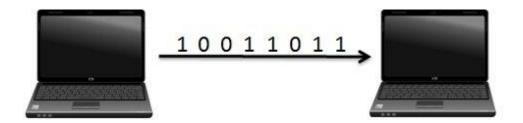


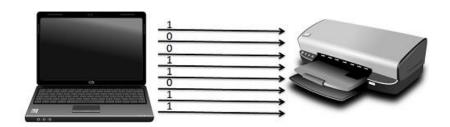
- El receptor podría enviar un NACK al detectar error para estimular la retransmisión antes de que expire el temporizador del emisor
- El emisor y receptor mantienen una ventana

3.5 EJEMPLOS DE PROTOCOLOS DE ENLACE DE DATOS

Transmisión serie y paralelo

- Transmisión serie. Los bits se transmiten a través de un único camino
- Transmisión paralelo. Los bits se transmiten en forma simultánea a través de varios caminos
- En telemática se estudia solamente la transmisión serie







Muestreo de la línea de transmisión

- En la transmisión de datos es importante la sincronización de los relojes del emisor y receptor
- El receptor debe saber la velocidad a la que se están transmitiendo los datos y en qué momentos inicia un bit
- Esto le permite muestrear la línea a intervalos constantes de tiempo para leer los bits
- Para ellos se utilizan dos técnicas de transmisión:
 - □ asíncrona y
 - □ síncrona



Transmisión asíncrona

- La transmisión se realiza de carácter en carácter
- Cada carácter o byte se trata independientemente
- El primer bit de cada carácter alerta al receptor de la llegada del carácter
- El % de bits suplementarios para alertar al receptor y validar el byte es alto
- La validación de cada carácter se lo hace a través del bit de paridad
- La paridad puede ser par o impar



Transmisión síncrona

- La transmisión se realiza en bloques de tramas
- El % de bits suplementarios para sincronizar y validar tramas es mucho menor
- Por tal razón la transmisión es más eficiente



Líneas síncronas y asíncronas

- Línea síncrona. Línea telefónica arrendada o dedicada. Siempre está a disposición
- Línea asíncrona. Línea telefónica básica o dial-up. La línea podría estar ocupada

н

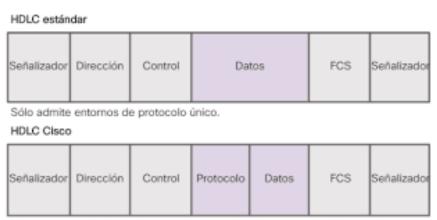
- Las redes WAN utilizan líneas punto a punto
- Las líneas punto a punto pueden ser de cobre o de FO
- Dos de los protocolos de enlace de datos para líneas punto-a-punto y puntomultipunto son:
- HDLC High-level Data Link Control
- PPP Point-to-Point Protocol, que es una mejora de SLIP



HDLC High-level Data Link Control

- Dentro de la carga útil usa relleno de bits (Sección 3.1.2 Entramado)
 Protocolo orientado a bit
- Se aplica en los datos insertando un 0 si hay 5 1s seguidos. Esto ayuda a la sincronización en la capa física
- Permite tramas con número fraccionario de bytes
- Usa transmisión con ventana deslizante y ACKs
- Cada trama empieza y termina con la bandera o señalizador 01111110
- La bandera permite además sincronizar los relojes de emisor y receptor



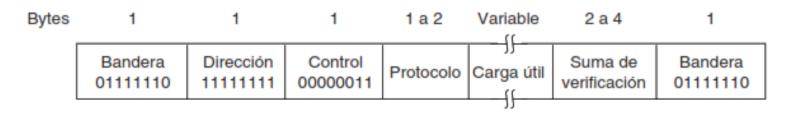


- Usa un campo de datos de protocolo para admitir entornos multiprotocolo.
- El campo de dirección identifica a la estación que recibirá la trama
- Este campo solo se necesita en transmisiones punto-multipunto (computador-terminales)
- 11111111 es una dirección de broadcasting. En este caso, el campo dirección identifica a la estación que ha transmitido la trama

.

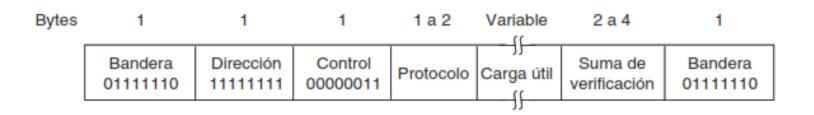
PPP Point-to-Point Protocol

- Es una mejora de SLIP Serial Line Internet Protocol
- Está orientado a bytes
- Las tramas tienen un número entero de bytes
- Dentro de la carga útil utiliza relleno de caracteres



- Solo se necesita una bandera entre tramas
- Si no hay tramas para enviar se envían una secuencia de banderas
- Dirección 11111111 indica que todas las estaciones deben aceptar la trama





- Control 00000011 indica tramas de control y son no numeradas
- Protocolo: IPv4, IPv6, IPX (Netware Novell), AppleTalk
- Carga útil de longitud variable. Longitud predeterminada 1500 bytes
- Usa Checksum para verificar la integridad de los datos (no usa CRC; este se lo usa en capas superiores)
- PPP es usado con muchos tipos de capas físicas