

ii'Oops!!

Tecnología de las Comunicaciones

La Capa de
Enlace de Datos
ppt#9



Ing Marcelo Semeria

Data Link Layer

Puntos a Considerar

Servicios provistos por la capa

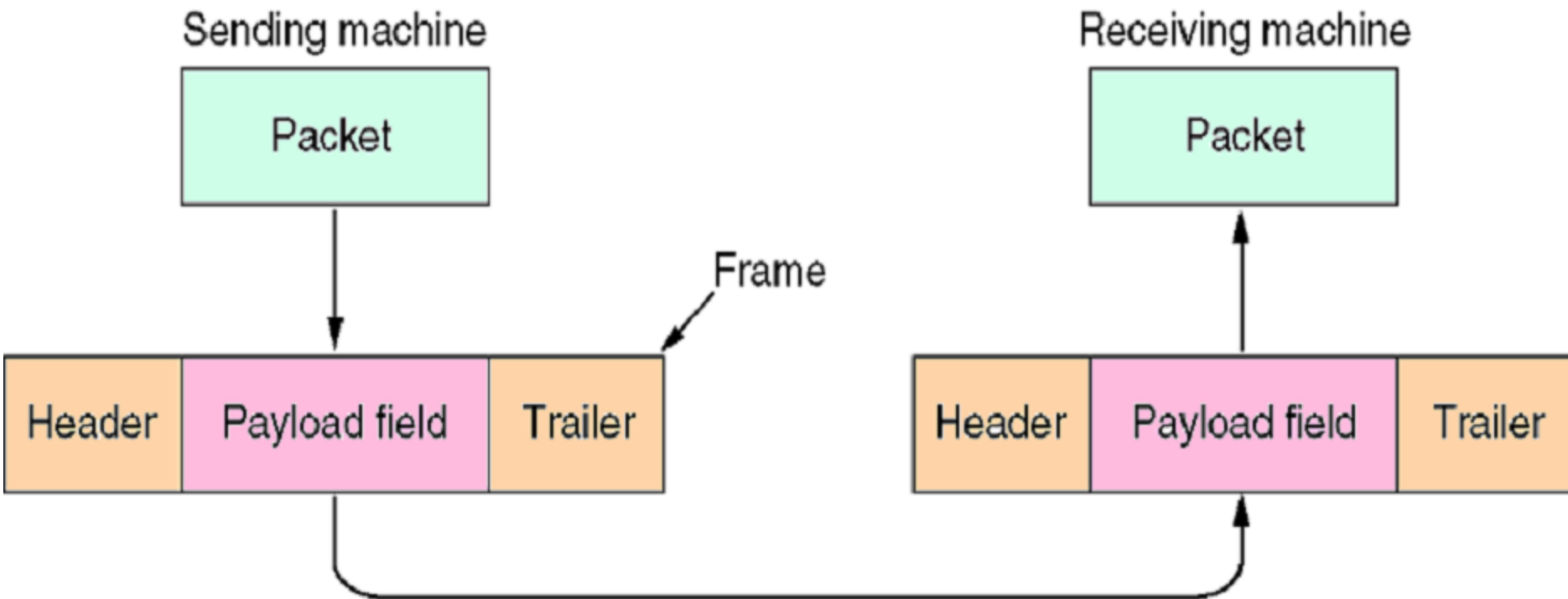
Armado de Tramas

Control de errores

Control de flujo

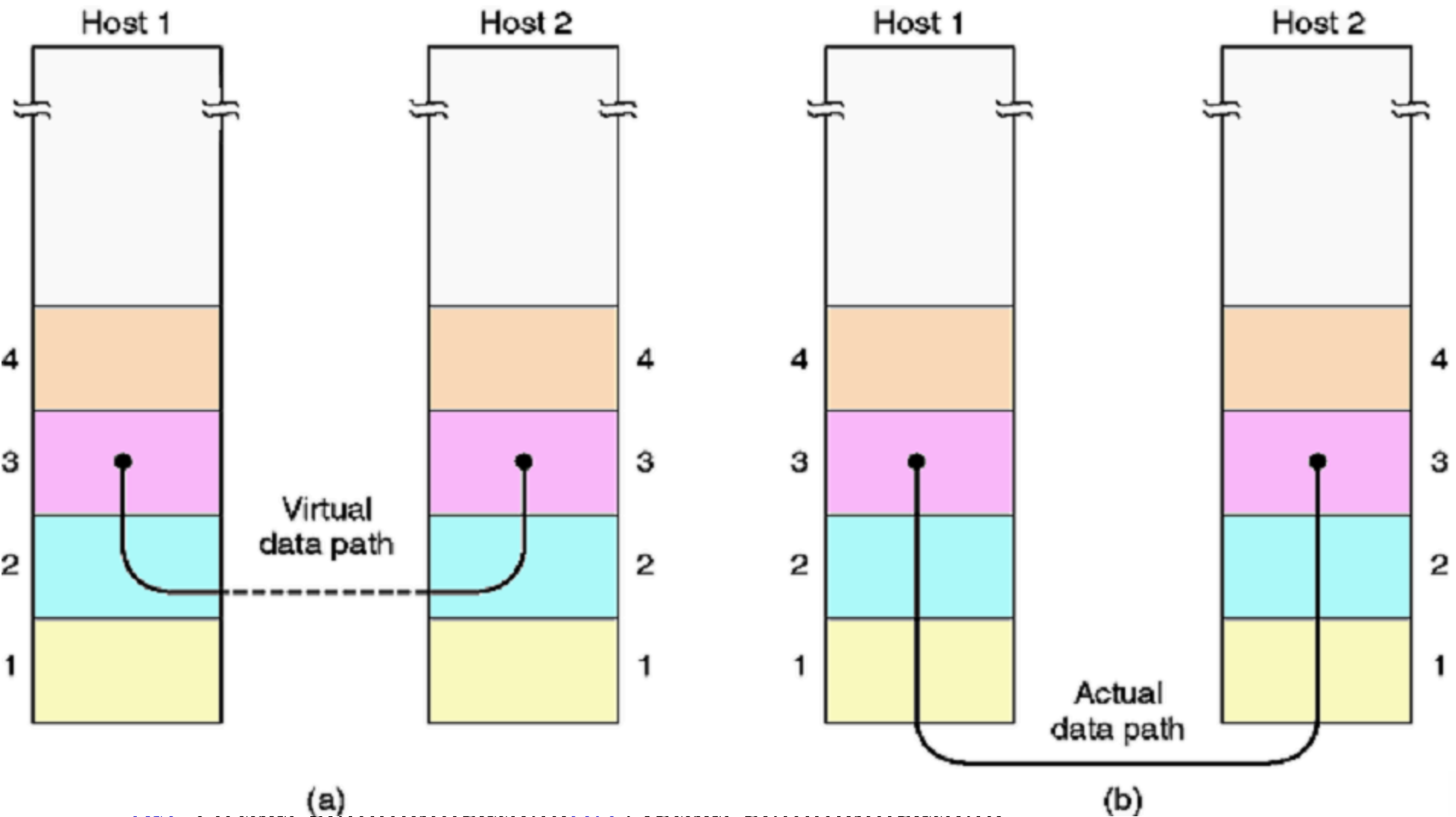


Funciones



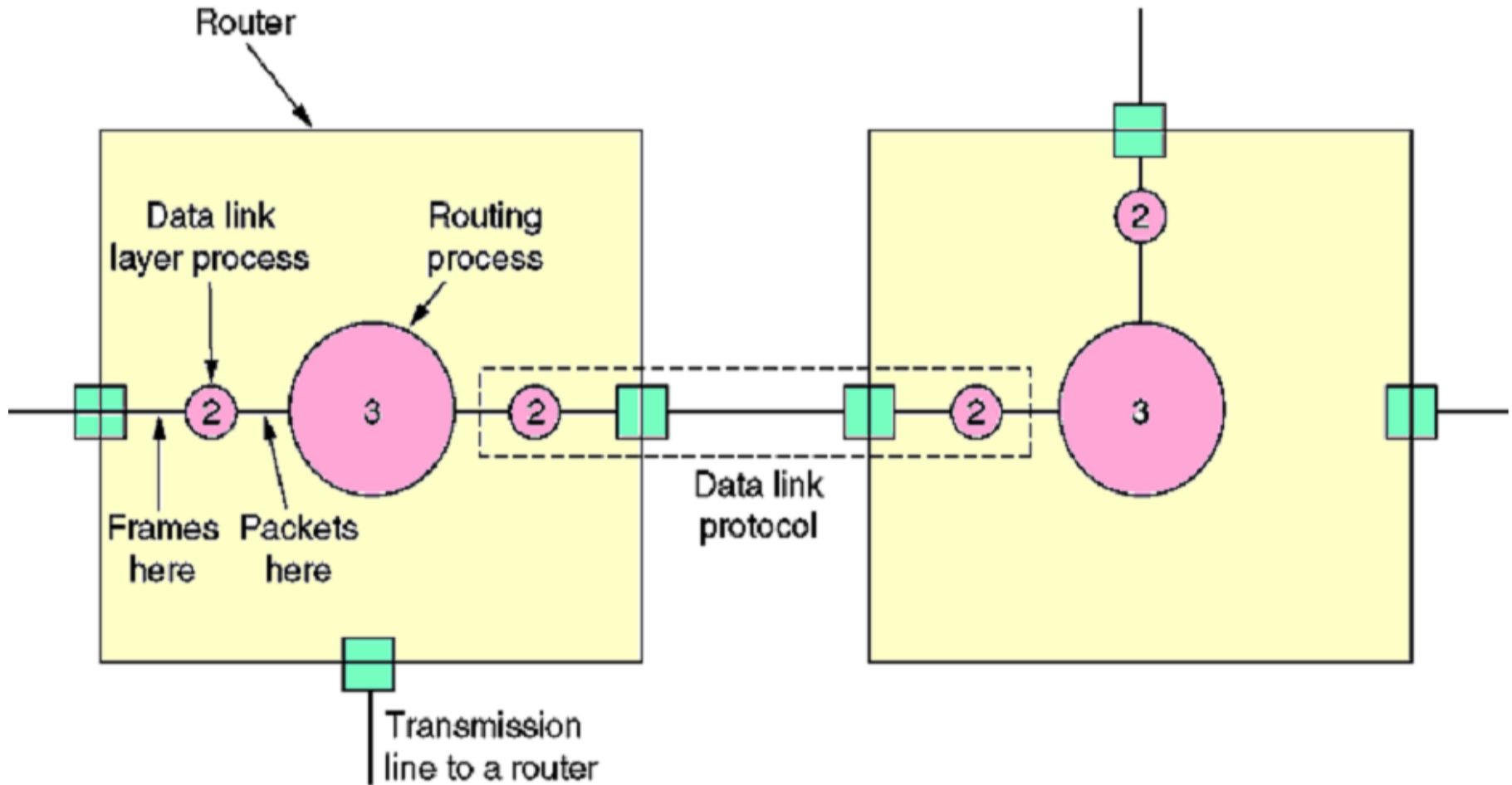
Relacion entre paquetes y tramas.

Servicios a capa de red

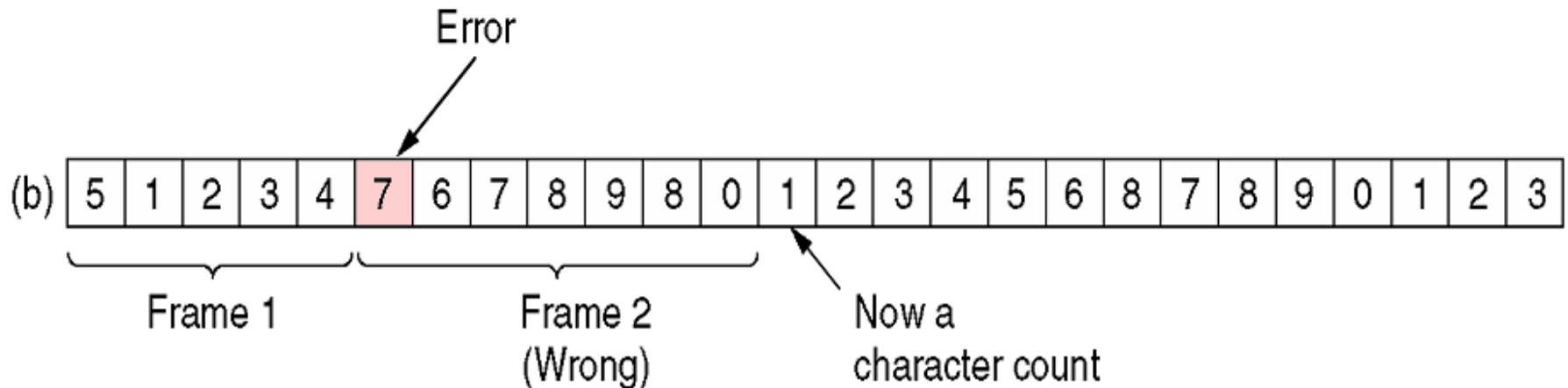
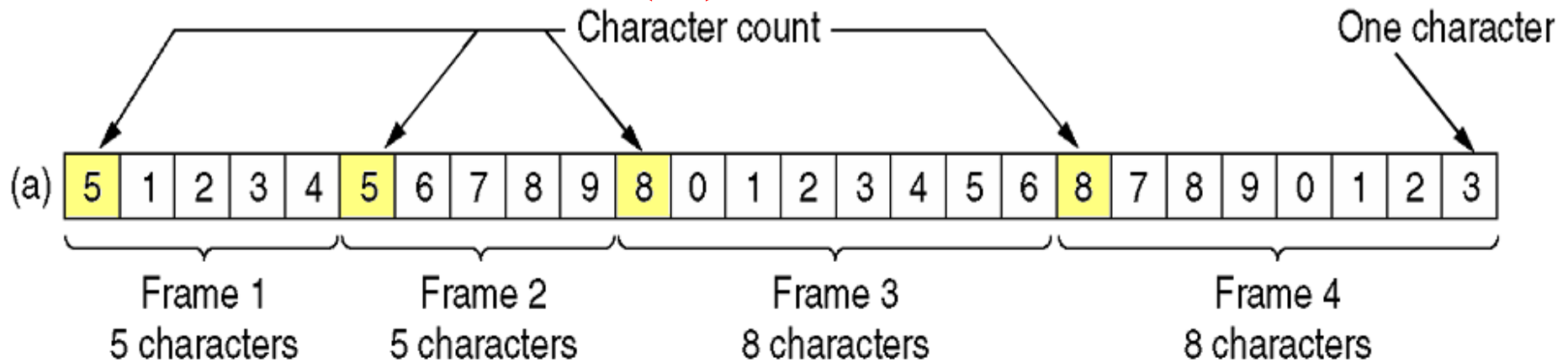


Servicios a Capa de Red (2)

Ubicacion de protocolos



Entramado (1)



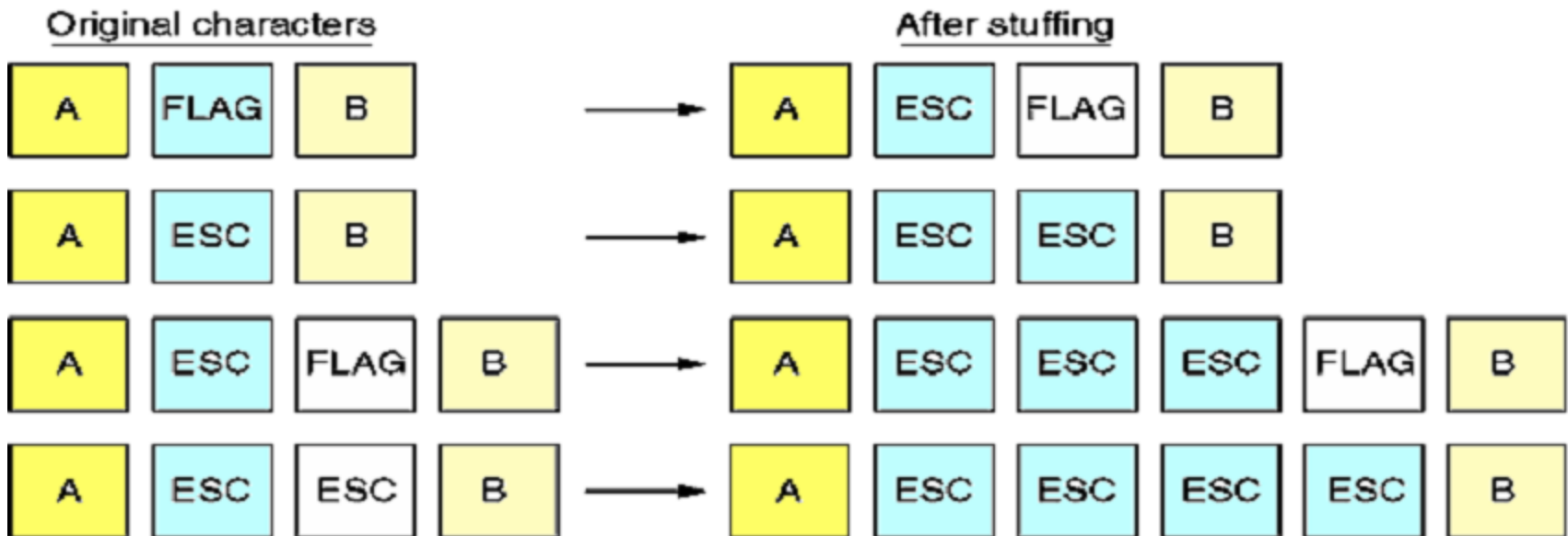
El flujo de caracteres

a) Sin Errores b) Con un error

Entramado (2)



(a)



(b)

(a) Trama delimitada por bits de flag.

(b) Cuatro ejemplos de secuencias de byte antes y despues del stuffing.

Entramado (3)

(a) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

(b) 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 0 0 1 0

The diagram illustrates the bit stuffing process. In row (b), the original data from row (a) is shown with three '0' bits circled in red. These are the stuffed bits. Blue brackets above the sequence identify three groups of five consecutive '1's: the first group is under the 7th to 11th bits, the second is under the 12th to 16th bits, and the third is under the 17th to 21st bits. Arrows point from the label 'Stuffed bits' to each of the three circled '0's. A blue bracket below the first five '1's is labeled '5 1's seguidos'.

(c) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

Bit stuffing

(a) Data original

(b) El dato como aparece en la linea.

(c) El dato como es almacenado en la memoria del receptor.

Errores

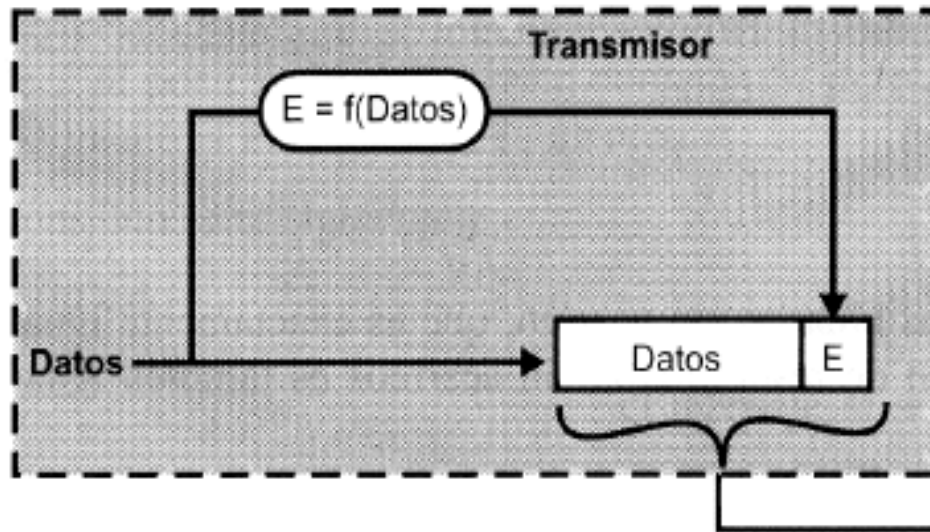
durante la transmisión
pueden aparecer errores



en la recepción hay que
comprobar si ha habido error

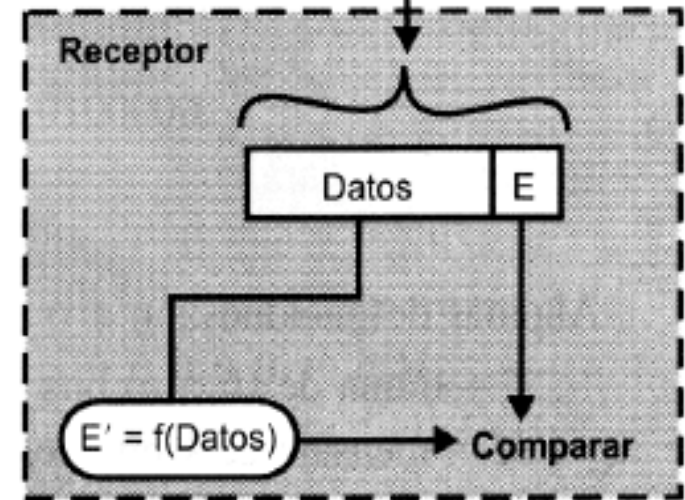
- Es necesario realizar, en el extremo receptor, una detección de errores
- La detección se realiza mediante códigos detectores de error:
 - Estos códigos incorporan a los datos, antes de ser transmitidos, información adicional que le sirve al receptor para saber si ha habido un error durante la transmisión

Códigos detectores de errores




- El receptor evalúa el código detector de error y determina si ha habido (o no) errores en la transmisión

E, E' = códigos de detección de errores
 f = función generadora del código de detección



Que hacer con un error?

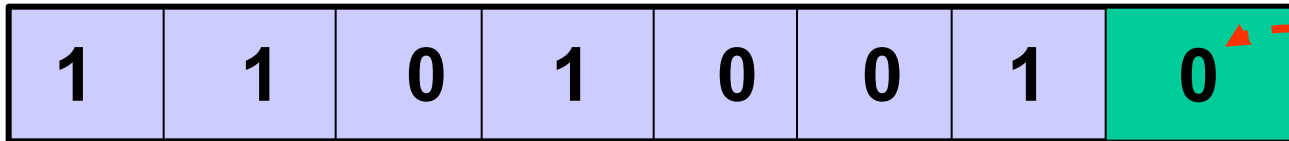
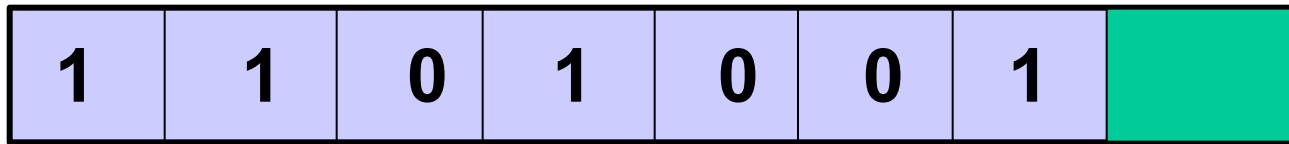
- Para poder corregir los errores se pueden emplear dos estrategias:

- 
- **FEC (Forward Error Correction)**: añade información de control que permitirá al receptor reconstruir la información correcta
 - **ARQ (Automatic Repeat Request)**: el receptor solicita al transmisor el reenvío de la información correcta

Control de paridad

Palabra a enviar **1101001**

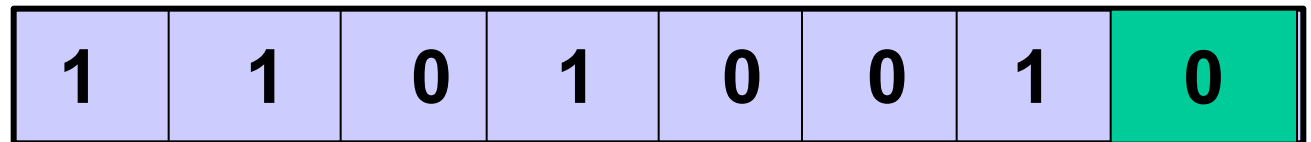
Bit de
pari
dad



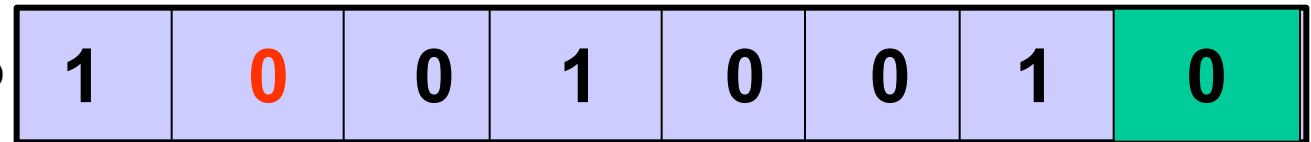
Se completa de forma de tener cantidad par de **UNOS**

TRANSMISION

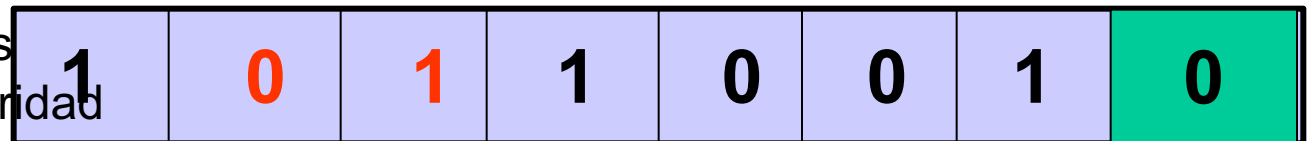
Todo OK



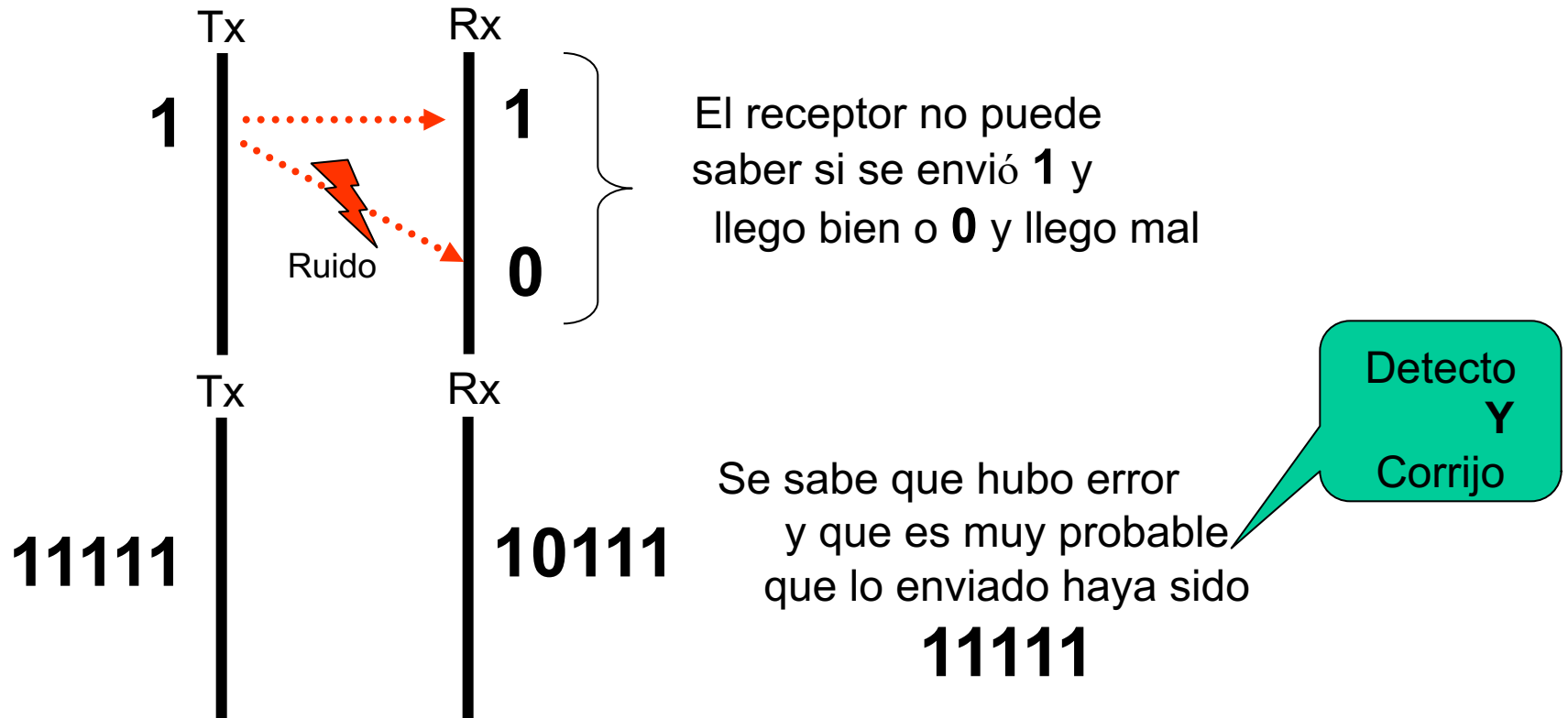
1 error detectado



2 errores **NO** detectados
pues mantiene paridad



Manejo de errores



El **precio** para detectar y corregir es **redundancia**:

- a) Tiempo de envío
- b) Cantidad de datos a enviar
- c) Tiempo de procesamiento

Error Detection and Correction

Codigos Correctores de error

Ej : Hamming

Codigos Detectores de error

Ej : CRC

Aplicación de Hamming (Generación)

Mensaje a transmitir : **10110** $m = 5$

Bits de redundancia $r = 4$ (Ver tabla a)

A) Se elijen como redundancia los bit potencia de 2 (en verde)

		1		0	1	1		0
1	2	3	4	5	6	7	8	9

B) Se completan los bit de redundancia para obtener **paridad par** en los bit controlados según **tabla b**.

Ej: **Bit 1** controla bits 1, 3, 5, 7, 9 \square X 1 0 1 0 \square x=0

Bit 2 controla bits 2, 3, 6, 7 \square X, 1, 1, 1 \square x=1

0	1	1	0	0	1	1	0	0
1	2	3	4	5	6	7	8	9

Se transmite la palabra: **011001100**

Tabla a

r	$m+r+1 \leq 2^r$	
a)	7	2
b)	8	4
c)	9	8
d)	10	16

Cumple

Tabla b

1	1
2	2
3	1,2
4	4
5	1,4
6	2,4
7	1,2,4
8	8
9	1,8

Decimales formados por potencias de 2

Aplicación de Hamming (Recuperación)

Mensaje recibido : **011011100** Del slide anterior con 1 error

A) El receptor verifica si hay errores, elije como redundancia los bit potencia de 2 (en verde)

0	1	1	0	1	1	1	0	0
1	2	3	4	5	6	7	8	9

B) El receptor verifica paridad con los mismos criterios que el transmisor

0	1	1	0	1	1	1	0	0
1	2	3	4	5	6	7	8	9

X	Ok		X				ok	
----------	----	--	----------	--	--	--	----	--

Bit 1 X : 1,3,5,7,9,
Bit 2 ok : 2,3,6,7,
Bit 4 X : 4,5,6,7,
Bit 8 ok : 8 9

Los bit comunes en los controles con error 5 y 7 Pero el control da como ok el 7

□ **Bit 5 Con ERROR**

La forma practica es sumar los pesos de los controles con error: **1 + 4 = 5** □

Bit 5

La palabra corregida es **011001100**

Hamming

Char.	ASCII	Check bits
H	1001000	00110010000
a	1100001	10111001001
m	1101101	11101010101
m	1101101	11101010101
i	1101001	01101011001
n	1101110	01101010110
g	1100111	01111001111
	0100000	10011000000
c	1100011	11111000011
o	1101111	10101011111
d	1100100	11111001100
e	1100101	00111000101

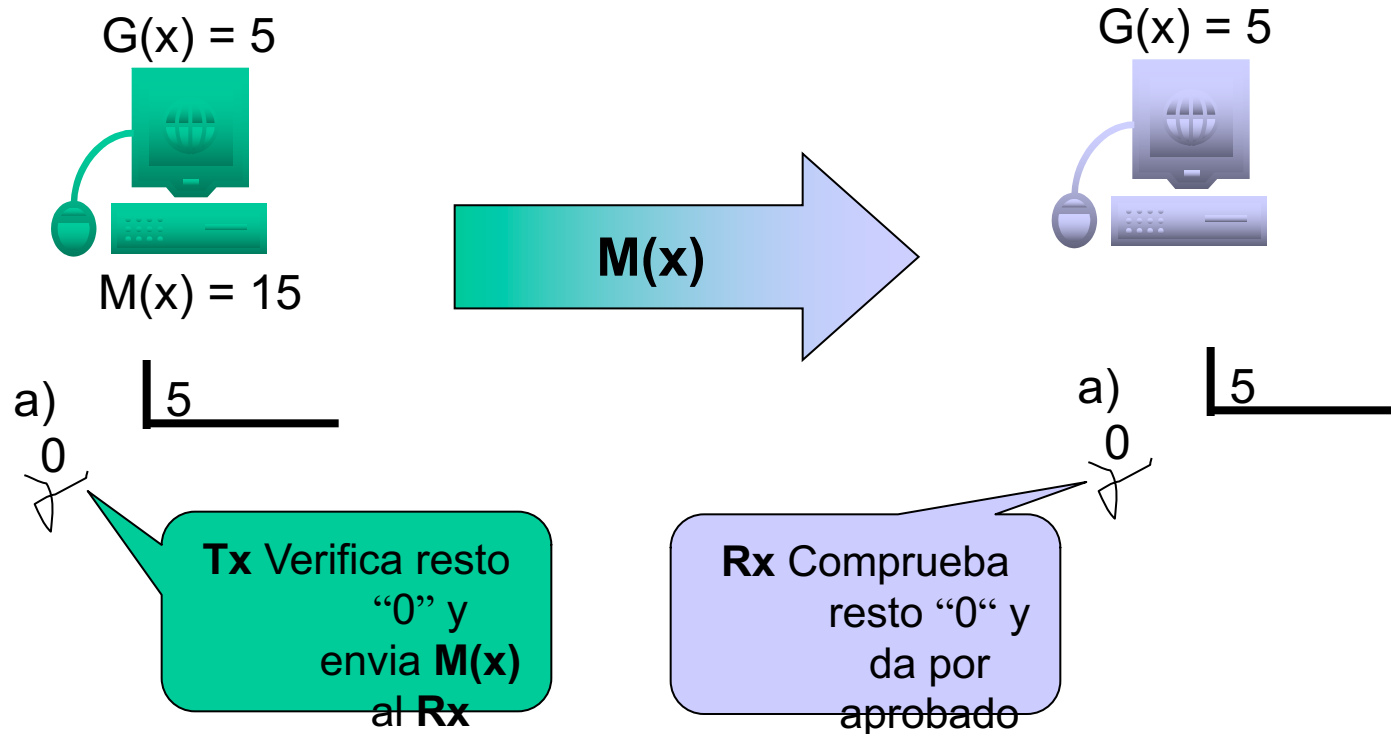
Order of bit transmission

Hamming

Mediante el agregado de redundancias se logra poder corregir errores

CRC Fundamento

Suponga que **Tx** y **Rx** comparten un mismo polinomio generador **G(x)**
Sea **M(x)** el mensaje a transmitir



Inconvenientes del método

- a) Se puede producir un error que mantenga resto cero (en el ej ~~15~~ 15 \square 25)
- b) $M(x)$ puede no tener resto cero con el $G(x)$ dado

CRC Fundamento

Suponga que **Tx** y **Rx** comparten un mismo polinomio generador **G(x)**
 Sea **M(x)** el mensaje a transmitir

$$G(x) = 5$$



Se soluciona eligiendo con cuidado el polinomio generador **G(X)**

Tx Verifica resto "0" y envia **M(x)** al **Rx**

M(x)

$$G(x) = 5$$

Se soluciona transformando **M(x)** en **M'(x)** divisible por **G(x)**, luego en el **Rx** se vuelve al mensaje original **M(x)**. (**M(x)**

$$22 \overset{R}{\underset{2}{\overline{)}}} \begin{array}{l} 5 \\ \hline \end{array} M'(x) \rightarrow 22 - 2 = 20 \text{ Divisible por } G(x)$$

Inconvenientes del método

- Se puede producir un error que mantenga resto cero (en el ej ~~15~~ 15 \square 25)
- M(x)** puede no tener resto cero con el **G(x)** dado

Codigos detectores de errores

Frame : 1 1 0 1 0 1 1 0 1 1

Generator: 1 0 0 1 1

Message after 4 zero bits are appended: 1 1 0 1 0 1 1 0 1 1 0 0 0 0

OR
Exclusive

El **resto** debe
reemplazar los ceros
agregados al
mensaje □ resto el
resto al mensaje.

Mensaje a Transmitir:
11010110111110

En el Rx luego de
verificar que no
tenga errores
(resto cero) se
eliminan los 4
últimos bits.


0 1 1 1 0
0 0 0 0 0
1 1 1 0
Remainder

CRC (en forma de polinomio)

- Divisor: $x^2 + 1$
- Datos iniciales: 110100111
- Añadimos tantos ceros como el grado del divisor: 11010011100

$$11010011100 = x^{10} + x^9 + x^7 + x^4 + x^3 + x^2$$

$$\frac{x^{10} + x^9 + x^7 + x^4 + x^3 + x^2}{x^2 + 1} = x^8 + x^7 + x^6 + x^4 + x + 1 + \frac{x + 1}{x^2 + 1}$$

Bits a añadir: $x + 1$  $x^{10} + x^9 + x^7 + x^4 + x^3 + x^2 + x + 1$

Información a transmitir: 11010011111

FIN ppt#9

Continuación

- a) Problemas de Tanenbaum
- b) Presentación de simulador a usar en próxima practica

