

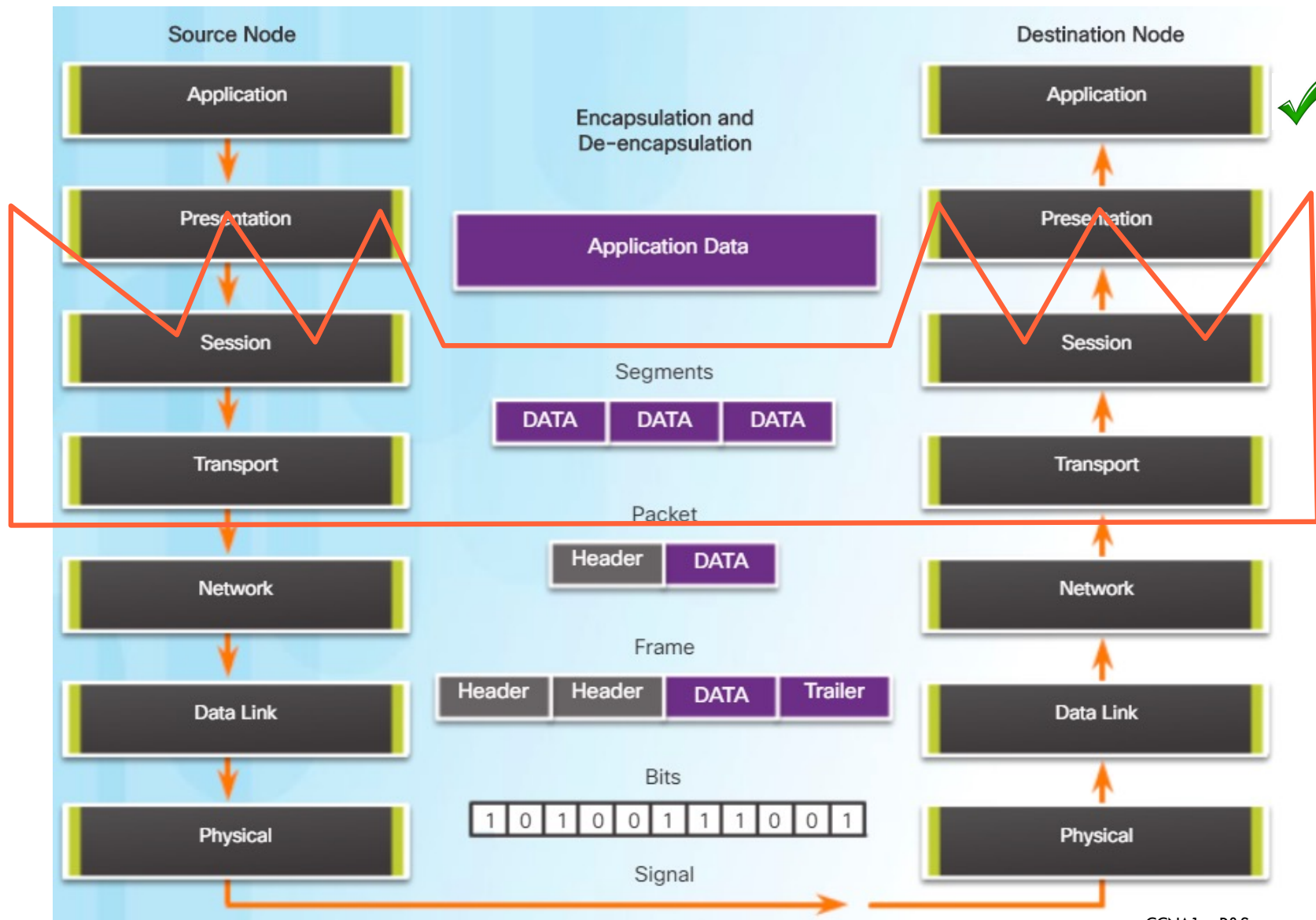


Redes de Computadores — RECO

Capa de Transporte

Ing. Claudia Patricia Santiago Cely

EL MODELO ...



FUNCIONES

➤ Conexión extremo a extremo

- NOAC
- OAC
 - Establecimiento de la conexión
 - Cierre de la conexión

➤ Recuperación de fallas

- Recuperación de conexiones
- Paquetes perdidos
- Cierre de conexión cuando espera paquetes
- Manejo de duplicados

FUNCIONES

- Direccionamiento
- Encapsulamiento
- Control de errores
- Control de Flujo
- Manejo de buffers
- Multiplexación
- Calidad de Servicio

SERVICIO DE LA CAPA DE TRANSPORTE

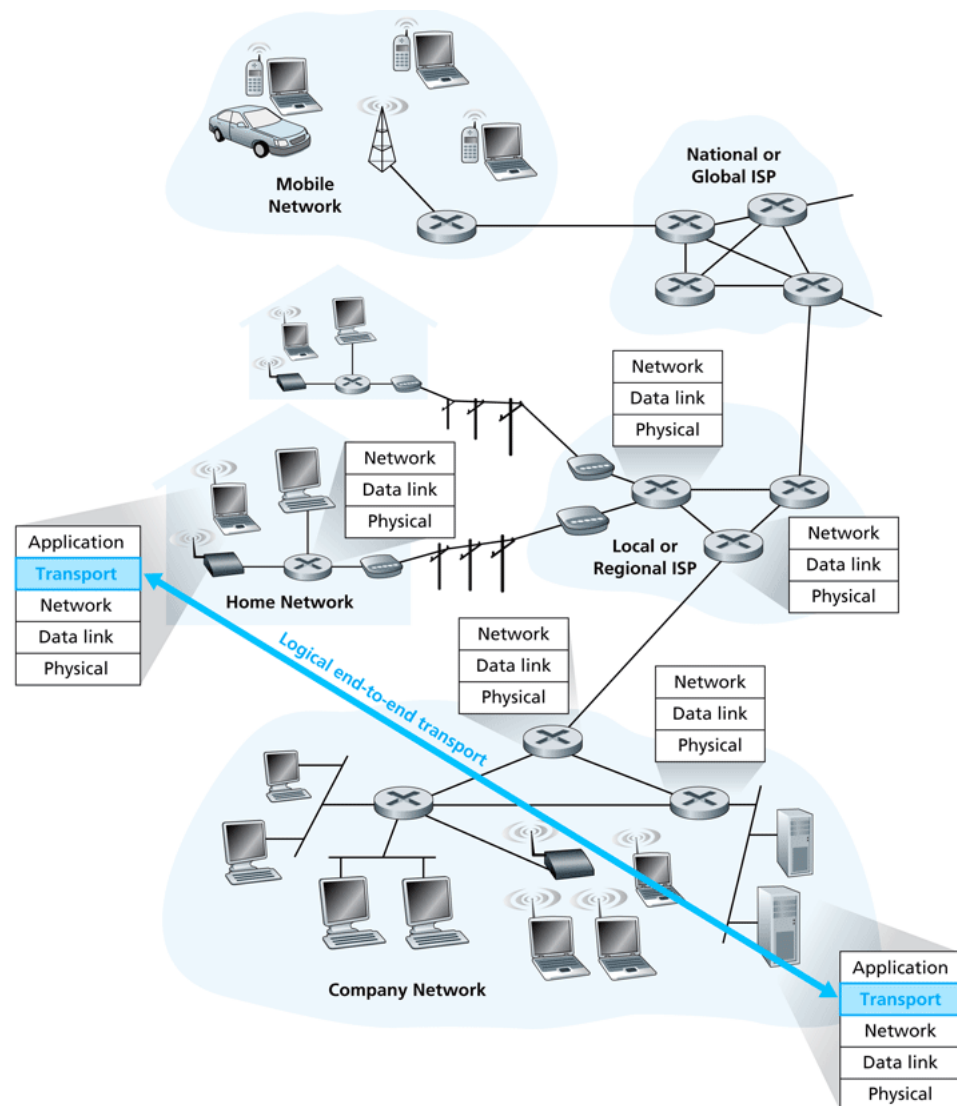


Figure 3.1 ♦ The transport layer provides logical rather than physical communication between application processes.



SERVICIO DE LA CAPA DE TRANSPORTE

Protocolos

- UDP
- TCP
- SCTP

SERVICIO DE LA CAPA DE TRANSPORTE

Características	UDP	TCP	SCTP
Generalidad	<u>User Datagram Protocol</u>	<u>Transmisión Control Protocol</u>	Stream Control Transmission Protocol
Direccionamiento	<u>Si</u>	<u>Si</u>	
Multiplexación	<u>Si</u>	<u>Si</u>	
Control de errores	<u>Opcional</u>	<u>Si</u>	
Control de flujo	No	<u>Si</u>	
Transmisión fiable Manejo de paquetes - Perdidos - Duplicados	No	<u>Si</u>	
Conexión	<u>NOAC</u>	<u>OAC</u>	
Encapsulamiento	<u>Si</u>	<u>Si</u>	
Control de congestión*	No	<u>Si</u>	
Temporizadores	No	<u>Si</u>	

CAPA DE TRANSPORTE EN INTERNET

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	UDP or TCP
Internet telephony	typically proprietary	UDP or TCP
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

REFERENCIAS

- CCNA Routing and switching: Introduction to Network and Routing and Switching Essentials modules. Cisco System. 2018.
- Wu, Chwan-Hwa, Introduction to computer networks and cybersecurity. CRC Press. 1336 páginas. 2013.
- Douglas E. Comer. Internetworking with TCP/IP Volume One (6th Edition). Pearson. 744 páginas. 2013.
- James Kurose and Keith Ross. Computer Networking: A Top-Down Approach (7th Edition). Pearson. 864 páginas. 2016.
- Stallings, William. Comunicaciones y redes de computadores. Prentice Hall, 7ma edición. 896 páginas. 2008.
- Computer Networks. 4/E. Andrew Tanenbaum.

PREGUNTAS



gracias

User Datagram Protocol

Servicio No orientado a Conexión

RFC 786

Funciones

- Multiplexación y demultiplexación
- Posible control de errores

Por qué usarlo

- La aplicación controla errores y orden de los datagramas
- Aplicaciones que transmiten pocos datos
 - Un solo datagrama
 - Pocos datos en el datagrama



PUERTOS

Punto de direccionamiento hacia el nivel de aplicaciones

- Administrados por el sistema operativo
- 65536

RFC 1700

- Well-known ports range from 0 through 1023. (Puertos bien conocidos)
 - Ej: 21=FTP, 80=WWW, 23=Telnet
- Registered ports are 1024 to 49151.
- Dynamic ports (also called private ports) are 49152 to 65535.

PUERTOS BIEN CONOCIDOS

Servicio	No
ftp-data	20
ftp	21
Telnet	23
SMTP	25
Time	37
Login	49
Domain	53
Bootsps	67
Bootpc	68
TFTP	69
Gopher	70
www-HTTP	80
POP3	110

Servicio	No.
IMAP	143
SNMP	161
SNMP	161
SNMPTRAP	162
BGP	179
LDAP	389
HTTPS	443
RIP	520
DHCP_client	546
DHCP_Server	547
IMAPS	993
POP3S	995
NFS	1023

PUERTOS REGISTRADOS

Servicio	No.
Oracle Database Client	2483
Secure Oracle Database Client	2483
Xbox LIVE and Games for Windows – Live	3074
MySQL database system	3306
PlayStation Network	3479 3480
PostgreSQL database system	5432
TeamViewer remote desktop protocol	5938
Apache Tomcat	8080
Microsoft Remote Administration for IIS Manager	8172
Bitcoin	8333
Kaspersky Network Agent	15000



MULTIPLEXACIÓN Y DEMULTIPLEXACIÓN

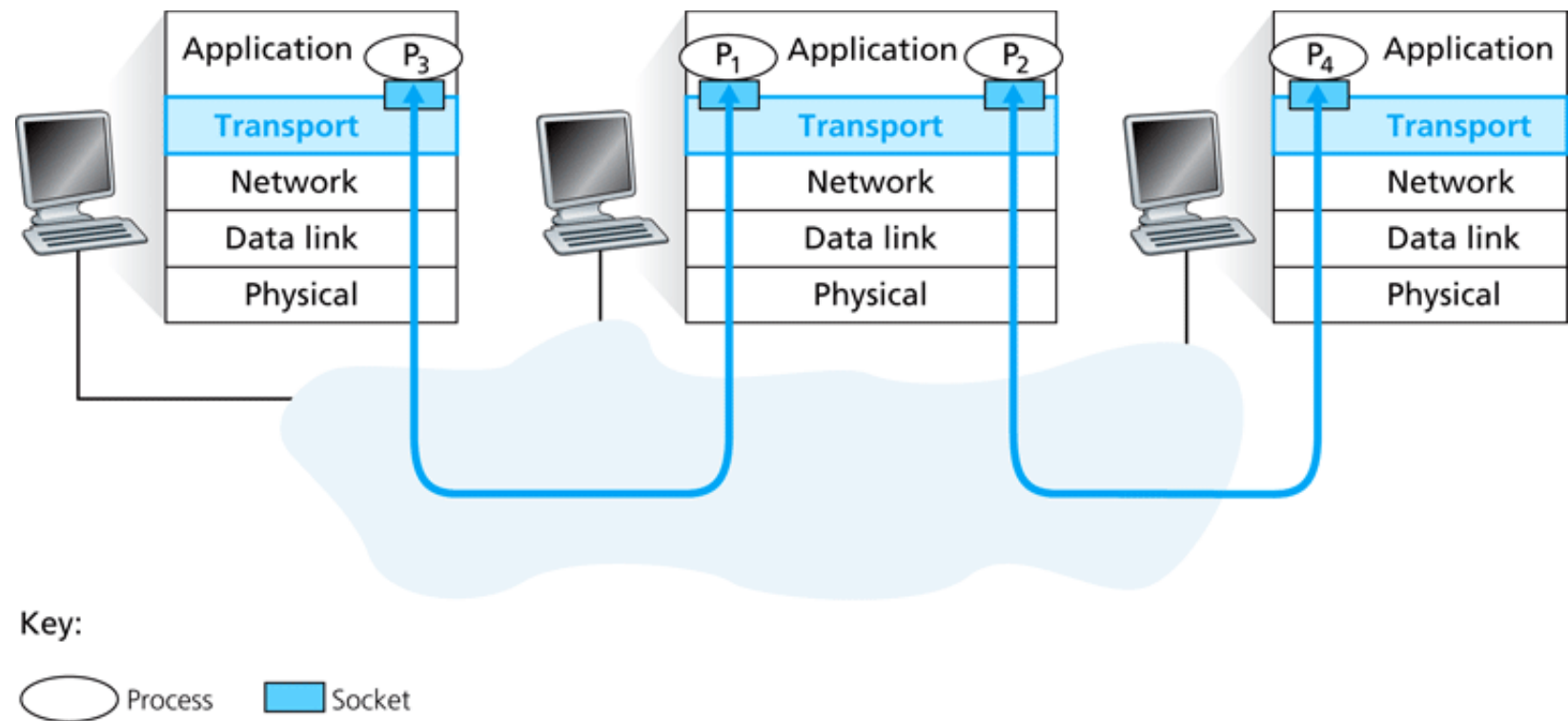
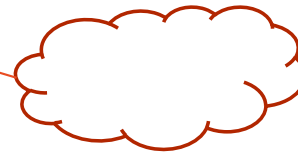


Figure 3.2 ♦ Transport-layer multiplexing and demultiplexing

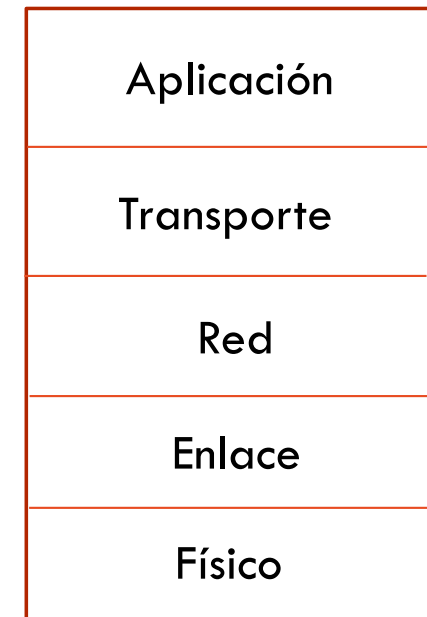
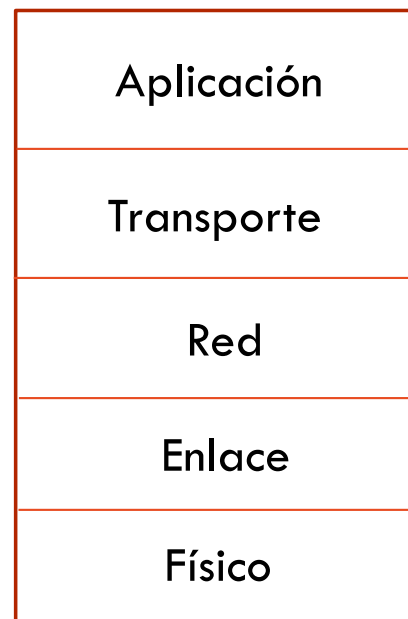
RELACIÓN APLICACIONES Y LA RED



❖ Procesos

❖ Sockets

❖ Puertos



MULTIPLEXACIÓN Y DEMULTIPLEXACIÓN

Sockets

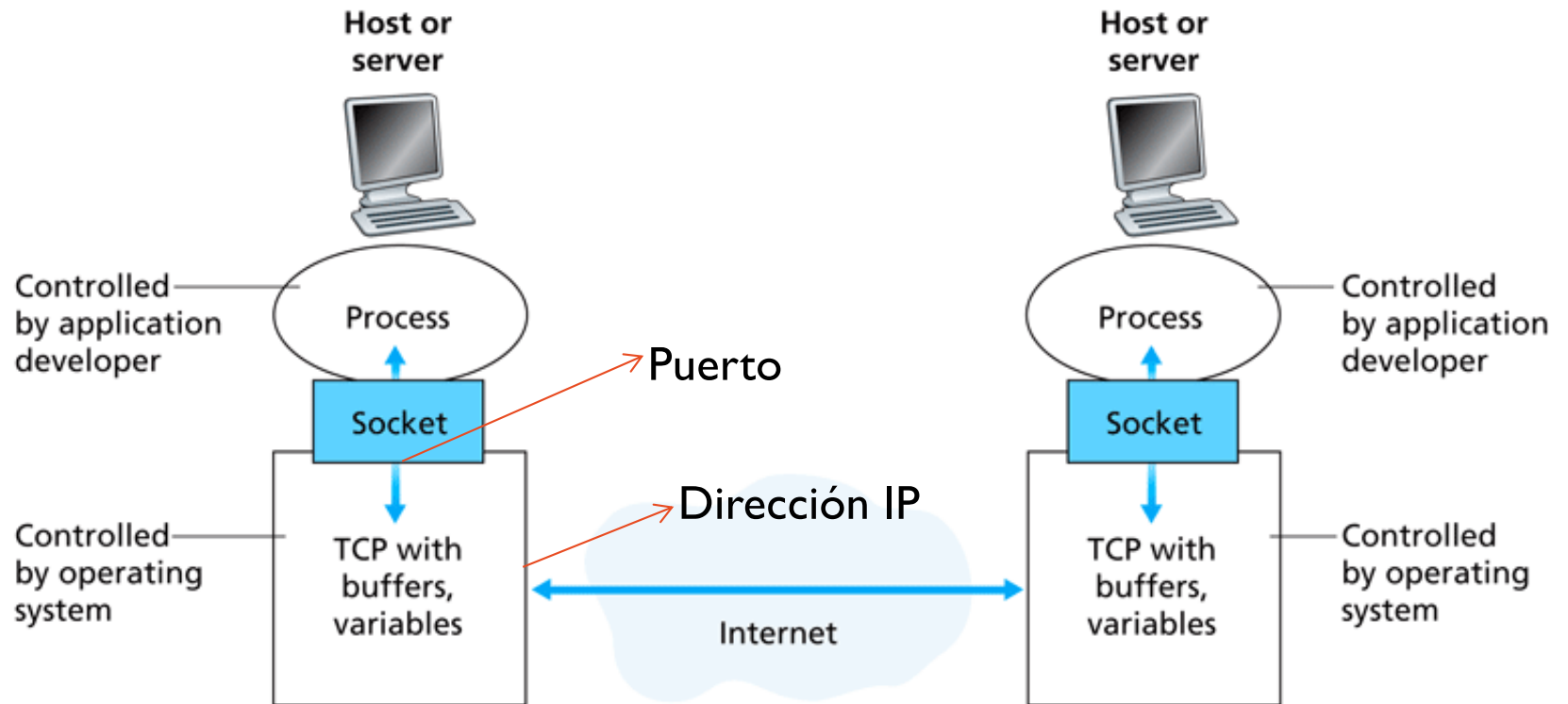


Figure 2.3 ♦ Application processes, sockets, and underlying transport protocol

MULTIPLEXACIÓN Y DEMULTIPLEXACIÓN

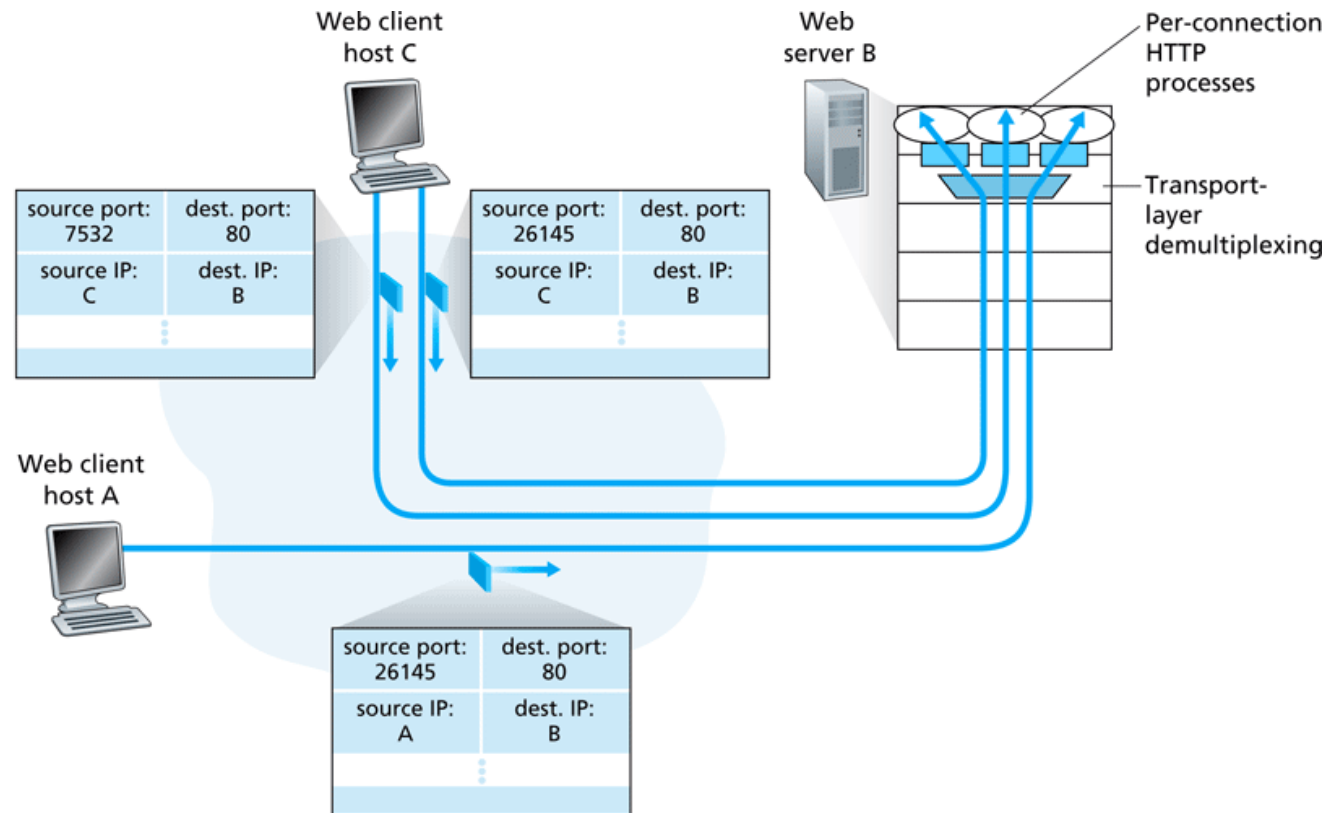
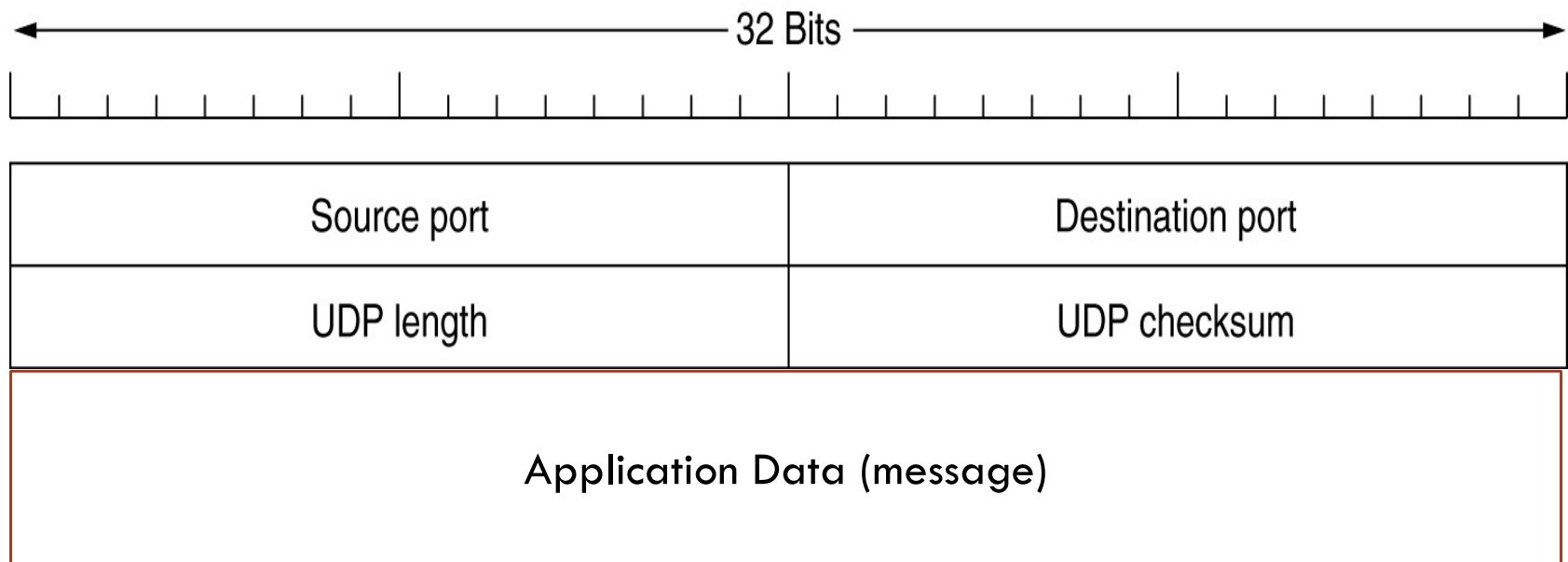


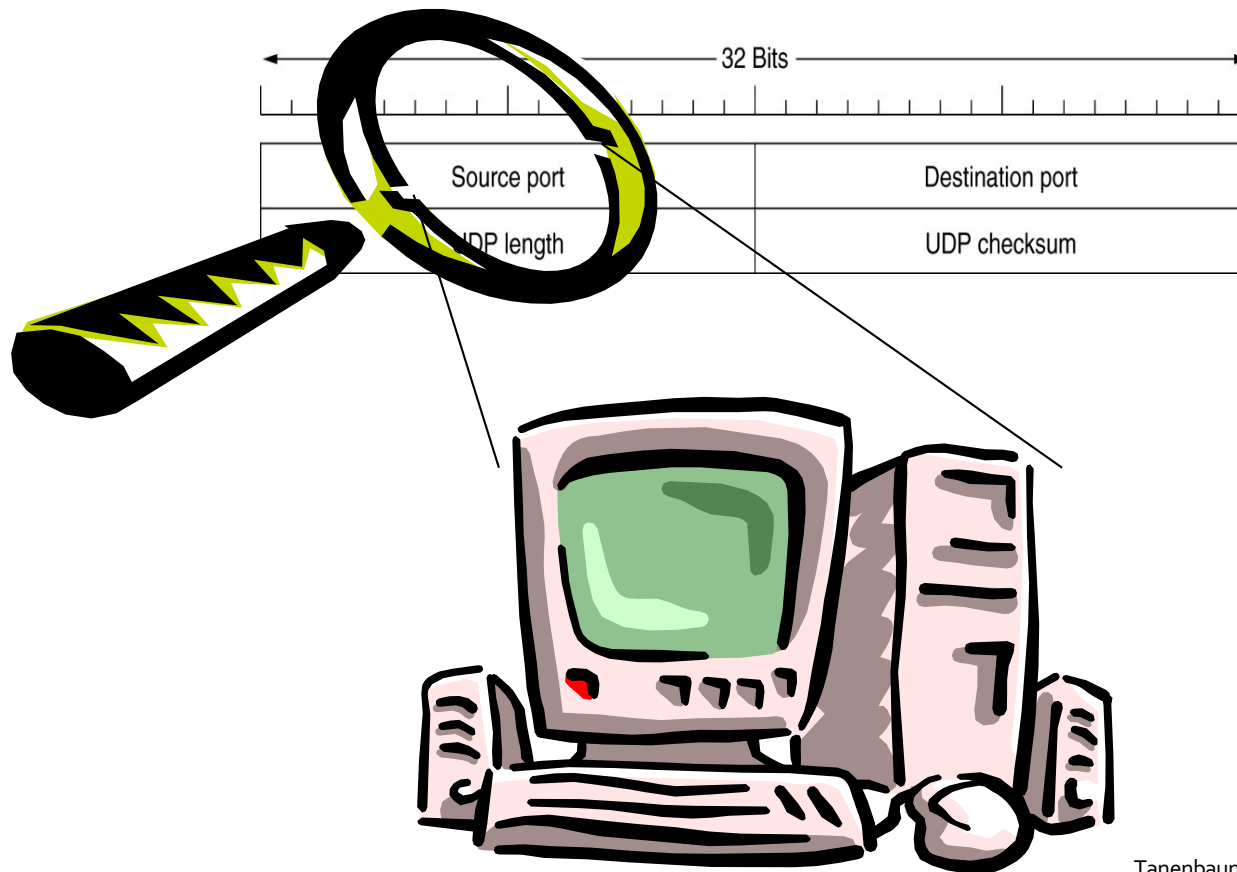
Figure 3.5 ♦ Two clients, using the same destination port number (80) to communicate with the same Web server application



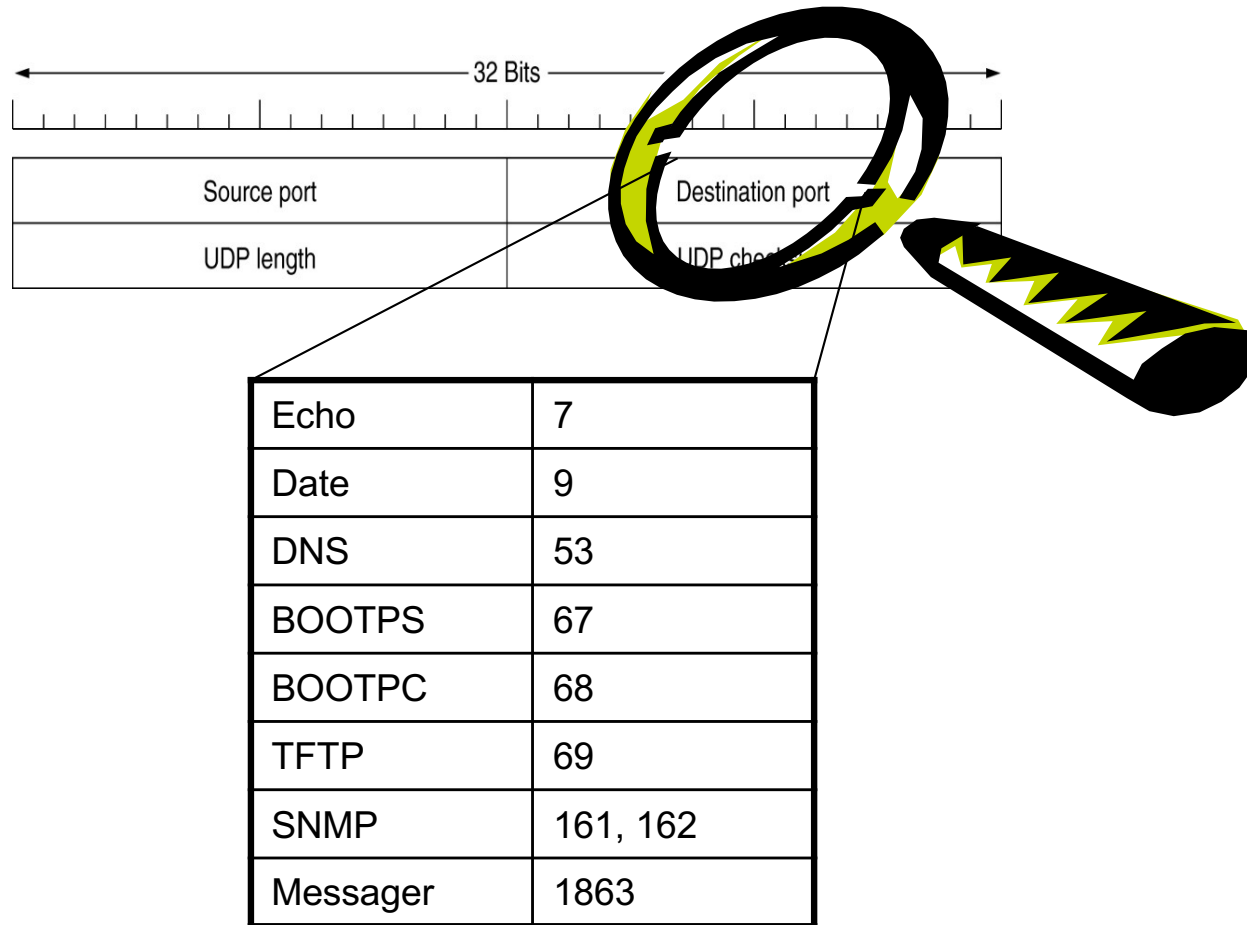
UDP ENCABEZADO



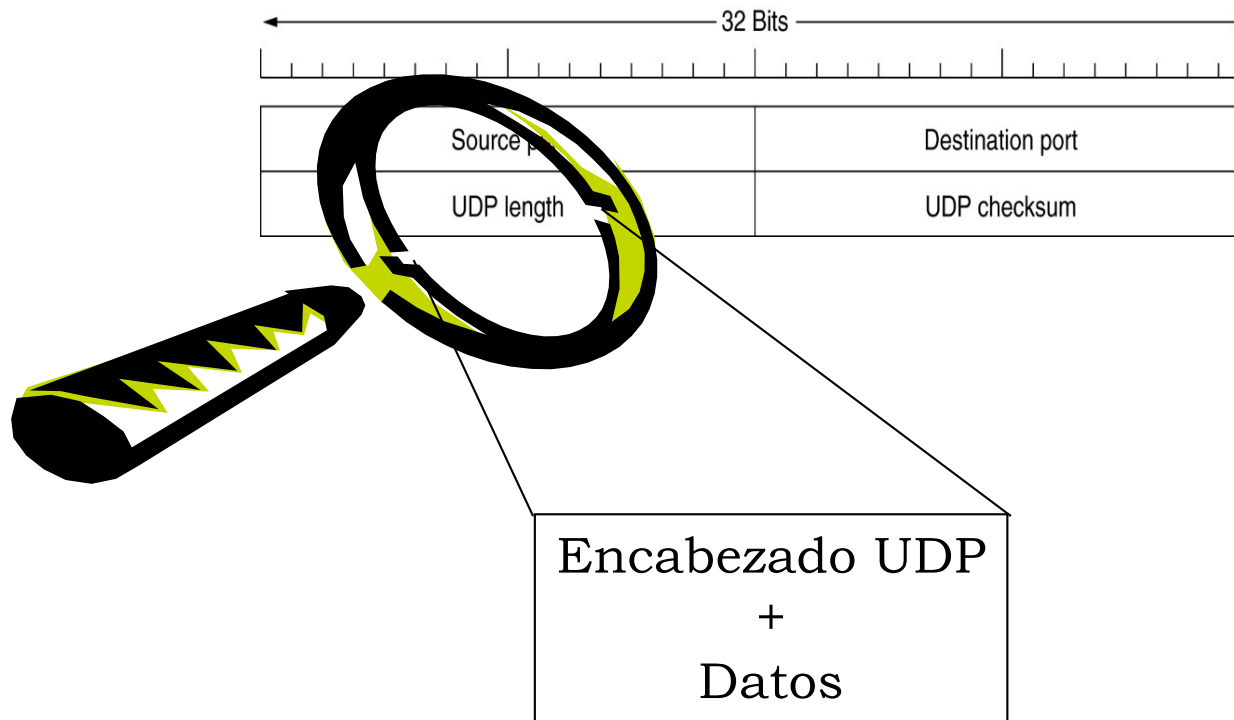
UDP ENCABEZADO



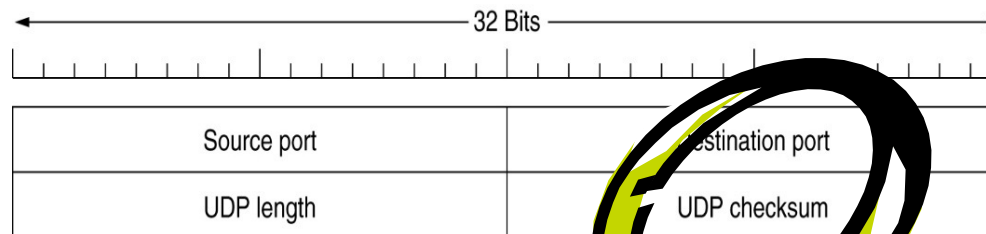
UDP ENCABEZADO



UDP ENCABEZADO



UDP ENCABEZADO



Opcional

0: no presente

ControlSegment: datos +

Seudoencabezado

Dirección IP origen		
Dirección IP destino		
0	Protocolo (=17)	Longitud UDP



CHECKSUM

- RFC 1071
- En origen
 - Completo a 1 de la suma de los datos en palabras de 16 bits
 - Ejemplo

```

0 1 1 0 0 1 1 0 0 1 1 0 0 0 0 0
0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
1 0 0 0 1 1 1 1 0 0 0 0 1 1 0 0
-----
0 1 0 0 1 0 1 0 1 1 0 0 0 0 0 1
                                     1
-----
0 1 0 0 1 0 1 0 1 1 0 0 0 0 1 0

```

```

0 1 1 0 0 1 1 0 0 1 1 0 0 0 0 0  1 0 1 1 1 0 1 1 1 0 1 1 0 1 0 1
0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1  1 0 0 0 1 1 1 1 0 0 0 0 1 1 0 0
1 0 1 1 1 0 1 1 1 0 1 1 0 1 0 1  0 1 0 0 1 0 1 0 1 1 0 0 0 0 0 1
                                     1
-----
                                0 1 0 0 1 0 1 0 1 1 0 0 0 0 1 0

```

complemento: 1 0 1 1 0 1 0 1 0 0 1 1 1 1 0 1 → Checksum

CHECKSUM

En destino

0	1	1	0	0	1	1	0	0	1	1	0	0	0	0	0	} ControlSegment
0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	
1	0	0	0	1	1	1	1	0	0	0	0	1	1	0	0	
<u>1</u>	<u>0</u>	<u>1</u>	<u>1</u>	<u>0</u>	<u>1</u>	<u>0</u>	<u>1</u>	<u>0</u>	<u>0</u>	<u>1</u>	<u>1</u>	<u>1</u>	<u>1</u>	<u>0</u>	<u>1</u>	} Checksum
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	
															1	
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

Si la operación incluye algún 0, el datagrama UDP llegó con errores.



ADMINISTRACIÓN DE LA CONEXIÓN

NOAC

No Orientada a conexión

- Transmite directamente al puerto que requiere los datos a través de un socket.
- Supone que destino está disponible.

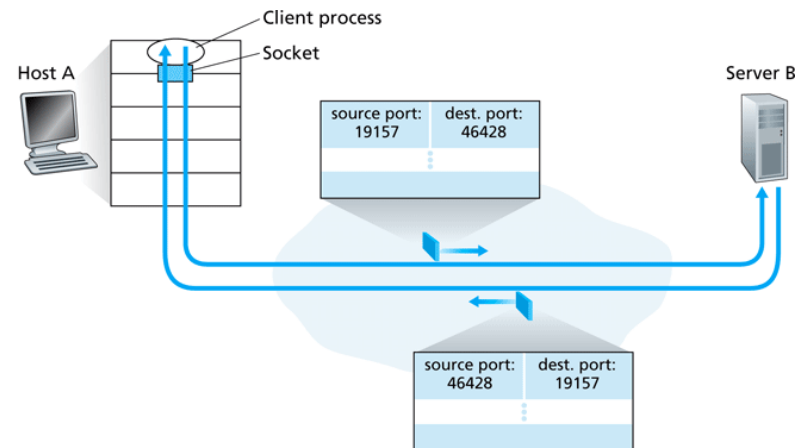


Figure 3.4 ♦ The inversion of source and destination port numbers

Transmission Control Protocol

Servicio Orientado a Conexión

RFC 793

Funciones

- Multiplexación y demultiplexación
- Control de errores
- Control de flujo
- Control de mensajes duplicados y perdidos
- Control de congestión

Por qué usarlo

- La aplicación requieren orden de entrega, transporte fiable.



CONTROL DE FLUJO

Se usa para que transmisores rápidos no ahoguen a receptores lentos

Generalmente los protocolos tiene implementadas una de las siguientes maneras para hacer este control de flujo

- Un momento en el que negocian la velocidad
- El receptor va indicando al transmisor cuando puede enviar mas mensajes.

Mecanismos:

- Simplex
 - Unrestricted (Canal perfecto)
 - For a noisy channel - Stop and wait
 - - Pueden dañarse datos
 - - Pueden perderse datos
- Sliding window
 - Go back n
 - Selective repeat



CONTROL DE FLUJO SIMPLEX

Sobre un canal perfecto

- Mas sencillo
- Transmisión de datos en una sola dirección
- No hay problema en el receptor para recibir datos. Buffers ilimitados.
- Nunca se pierden ni dañan datos

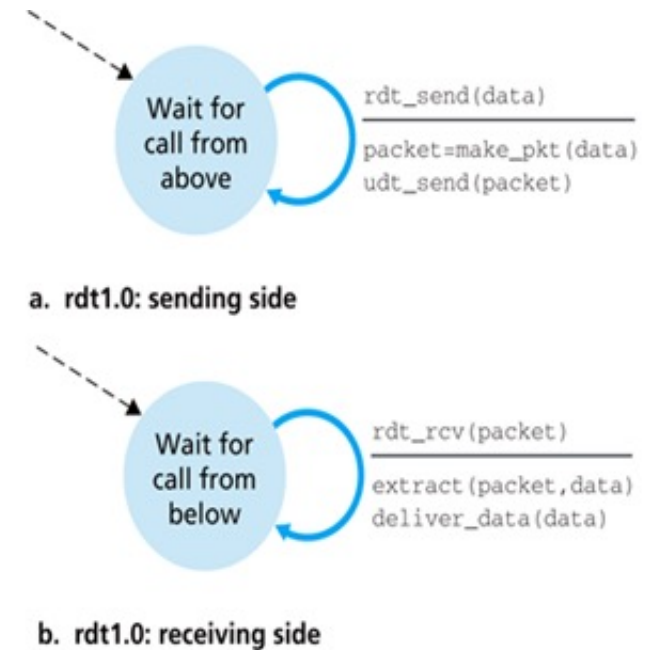


Figure 3.9 ♦ rdt1.0 – A protocol for a completely reliable channel

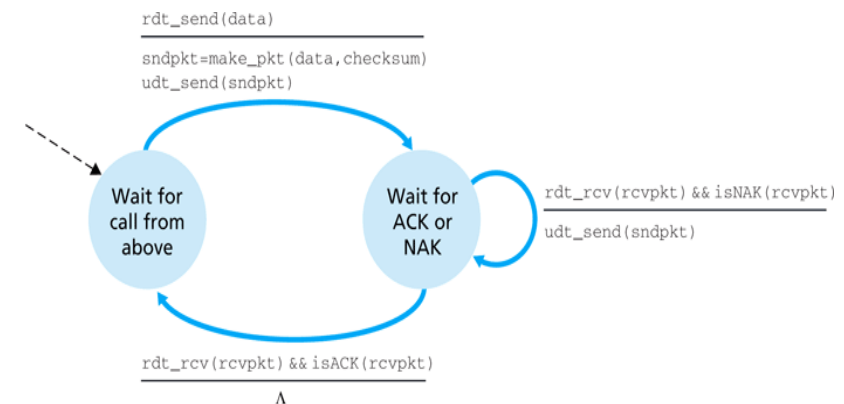
CONTROL DE FLUJO SIMPLEX

Sobre un canal ruidoso – Stop and wait (pueden dañarse datos)

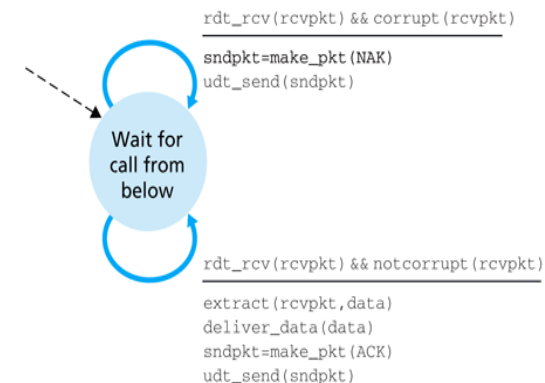
- Parada y espera



- Se pueden presentar errores durante la transmisión
- El receptor
 - Detección de errores: verifica que los datos que recibe se encuentren sin errores antes de pasarlos a la capa superior
 - Feedback: Envía mensaje a origen indicando el estado de los datos (ACK o NAK)
- El origen retransmite si se requiere



a. rdt2.0: sending side



b. rdt2.0: receiving side

Figure 3.10 ♦ rdt2.0—A protocol for a channel with bit errors

CONTROL DE FLUJO SIMPLEX

Sobre un canal ruidoso (pueden dañarse datos)

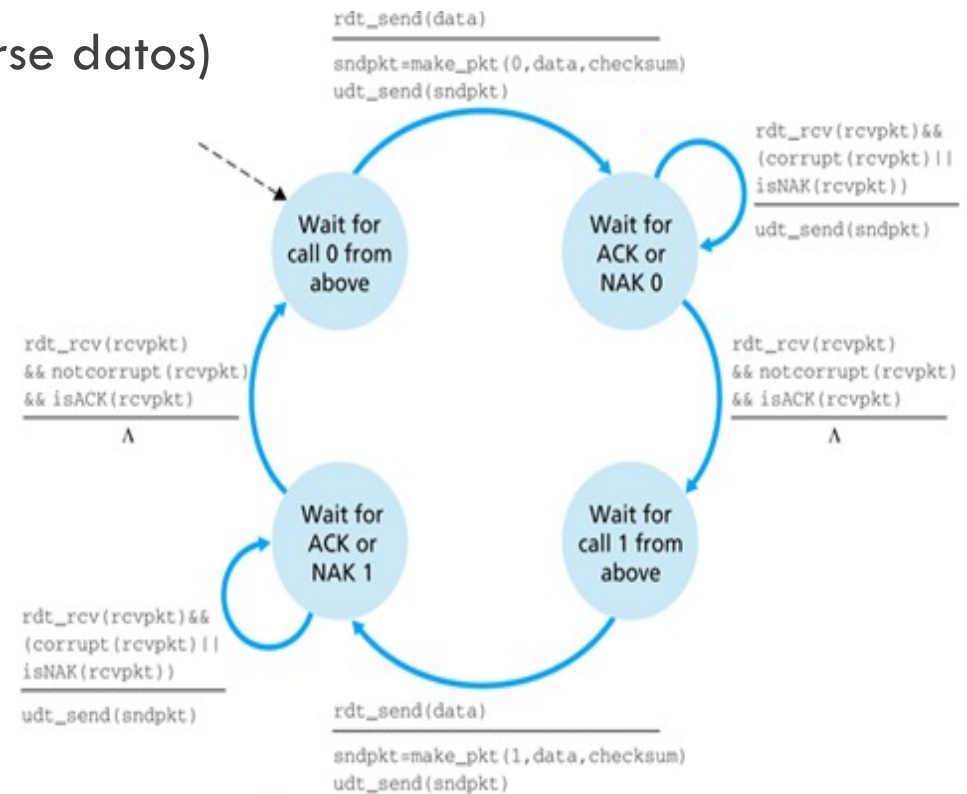


Figure 3.11 ♦ rdt2.1 sender

CONTROL DE FLUJO SIMPLEX

Sobre un canal ruidoso (pueden dañarse datos)

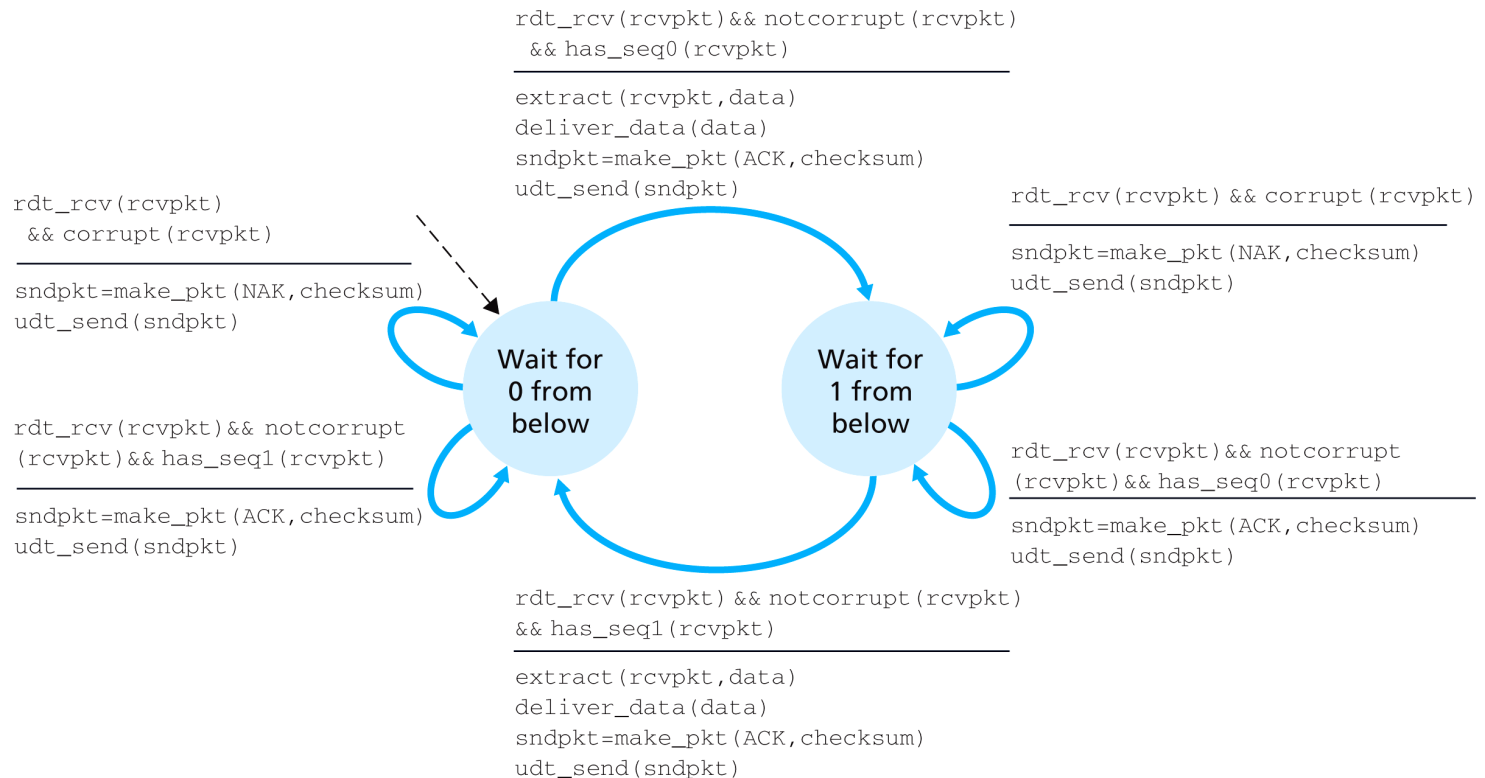


Figure 3.12 ♦ rdt2.1 receiver

CONTROL DE FLUJO SIMPLEX

SOBRE UN CANAL RUIDOSO (PUEDEN PERDERSE DATOS)



✦ Problemas:

✦ ¿Qué pasa si se pierde un mensaje?

✦ ¿Qué pasa si lo que se daña un mensaje de confirmación de recibo (ACK o NAK)?

✦ ¿Qué pasa si se pierde una confirmación de recibo?

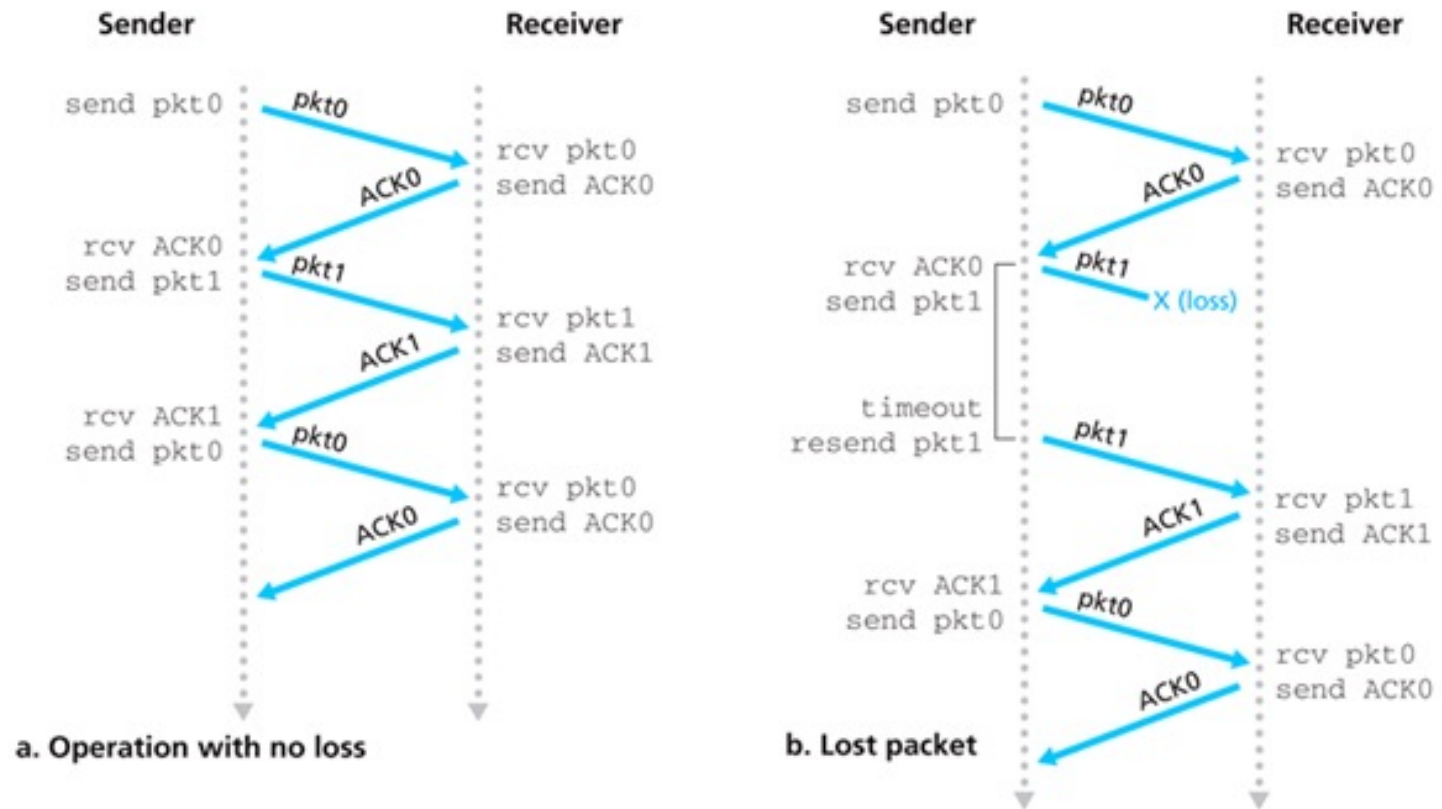
✦ Soluciones

✦ Números de secuencia(1 /0)

✦ Timeouts

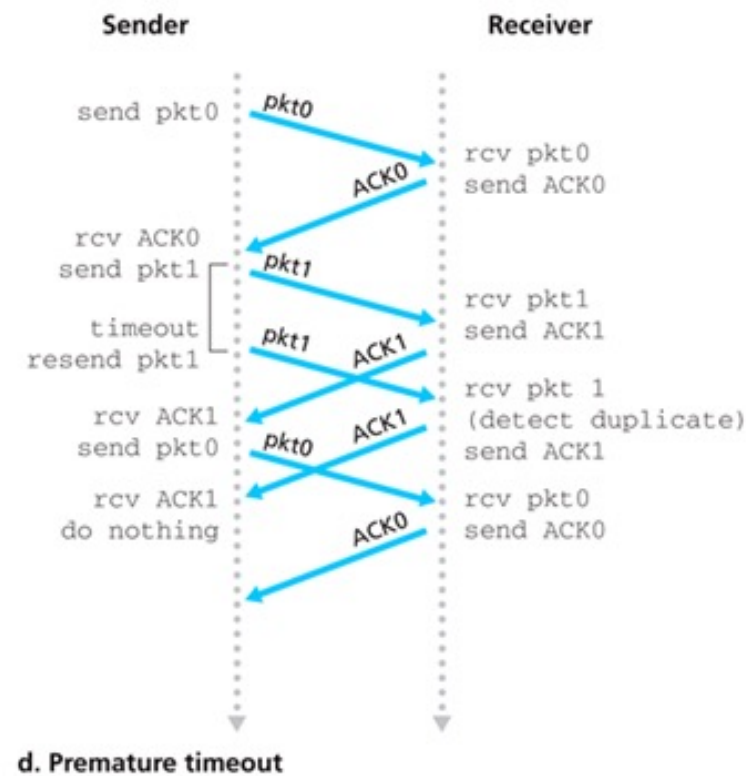
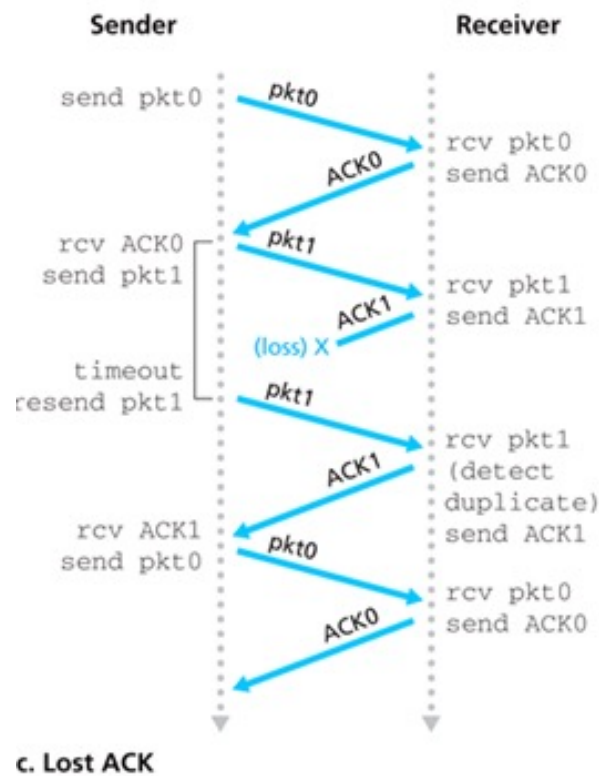
CONTROL DE FLUJO SIMPLEX

Sobre un canal ruidoso (pueden perderse datos)



CONTROL DE FLUJO SIMPLEX

Sobre un canal ruidoso (pueden perderse datos)



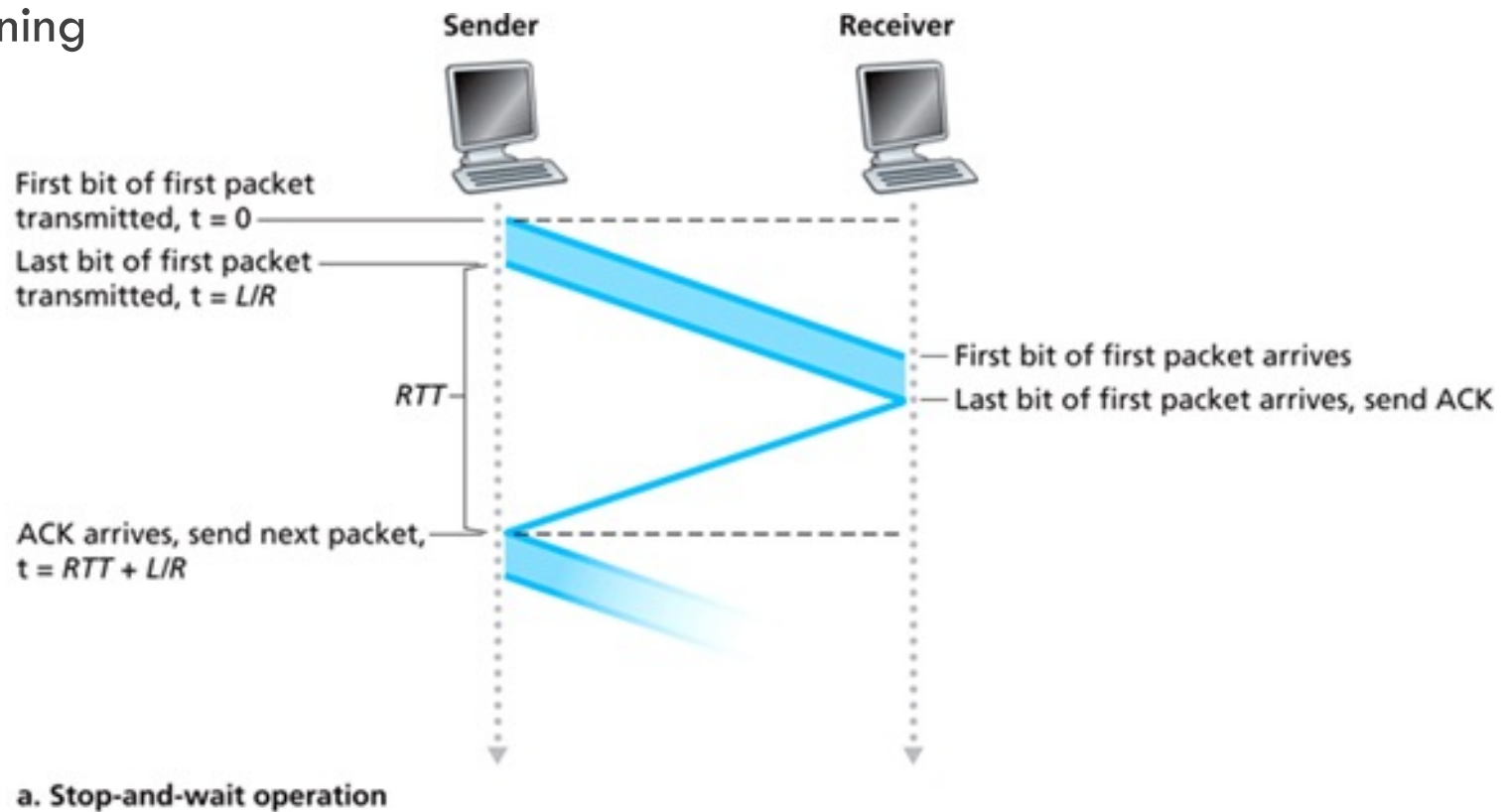
CONTROL DE FLUJO SLIDING WINDOW

- HALFDUPLEX/FULLDUPLEX
- PIPELINING
- PIGGY BACKING
- USO MÁS EFICIENTE DEL CANAL



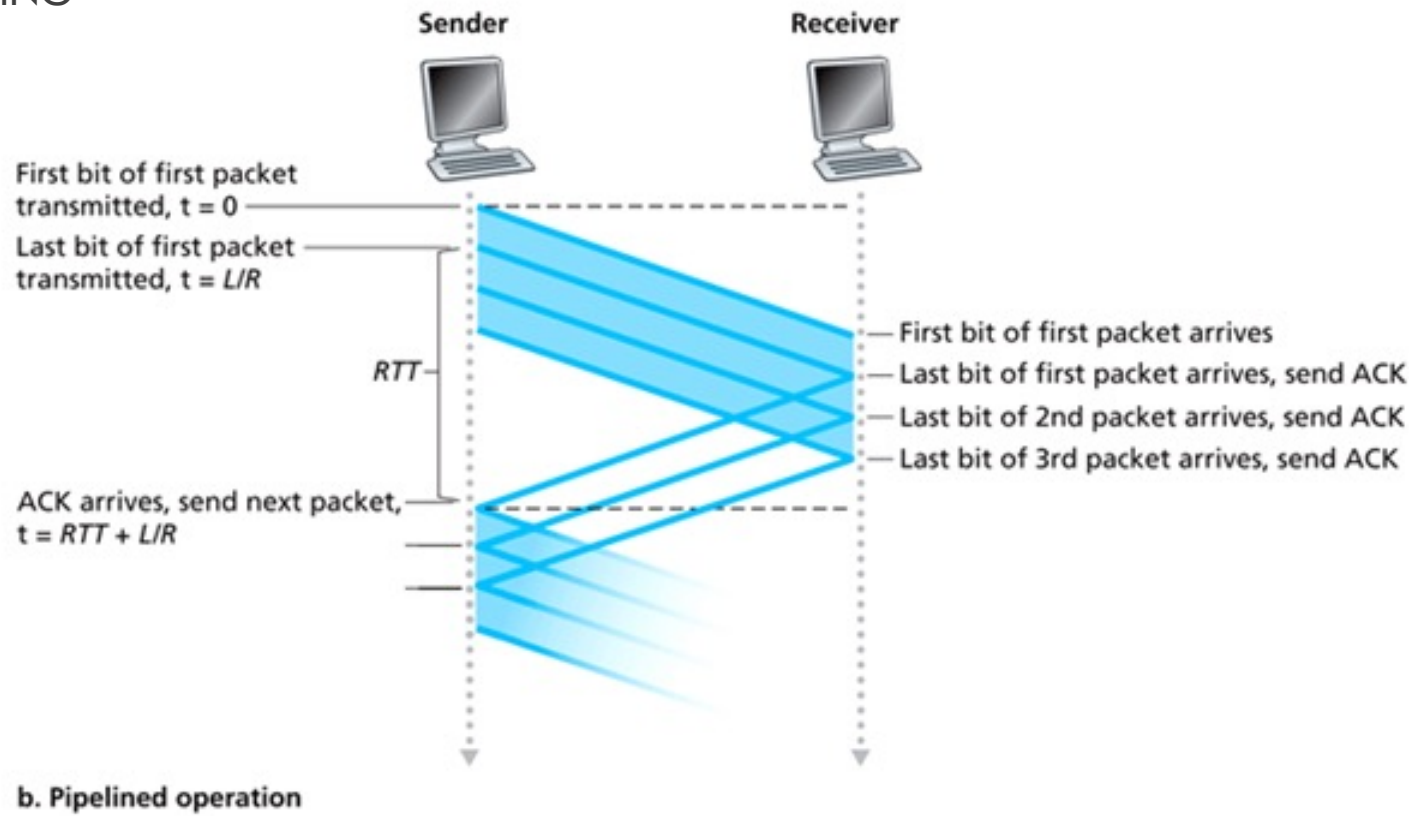
CONTROL DE FLUJO SLIDING WINDOWS

Pipelining



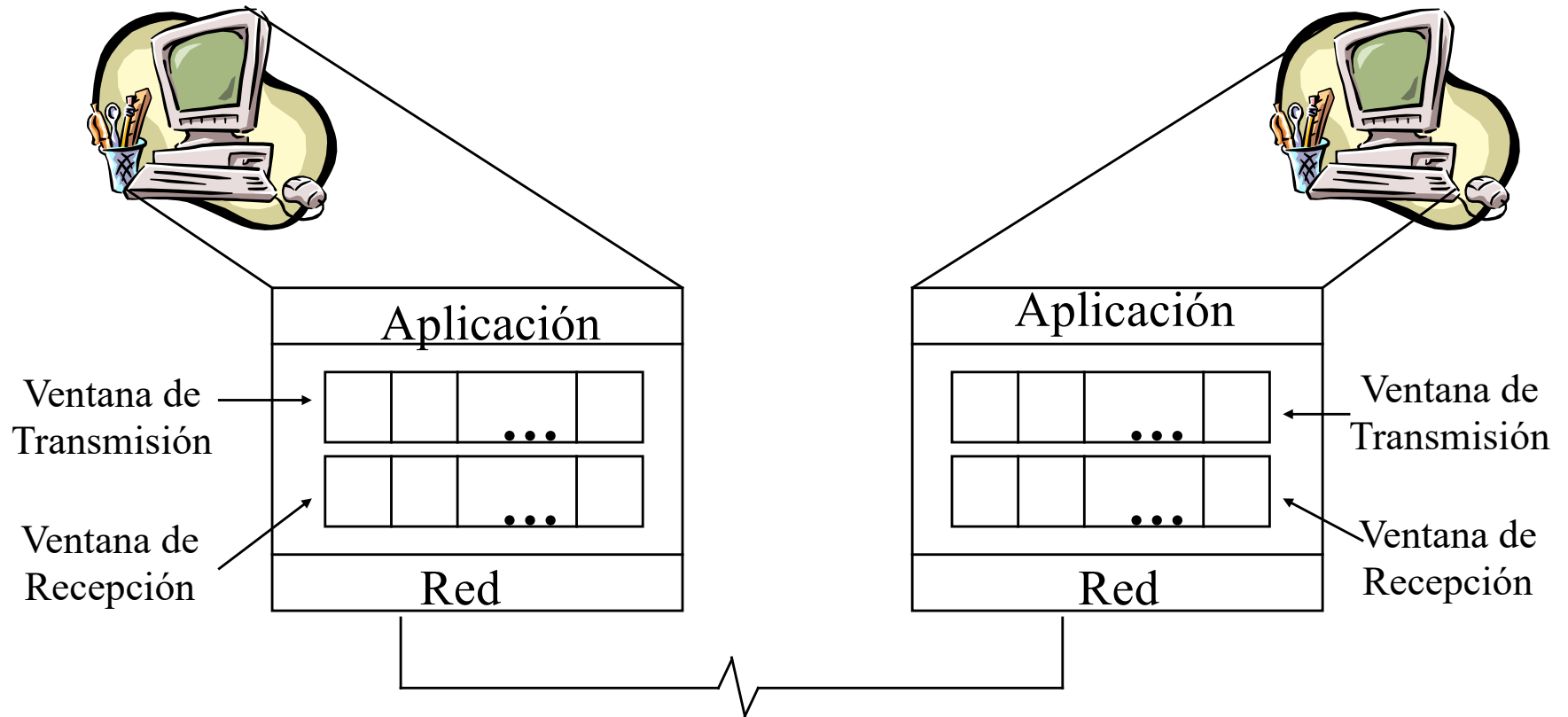
CONTROL DE FLUJO SLIDING WINDOW

• PIPELINING

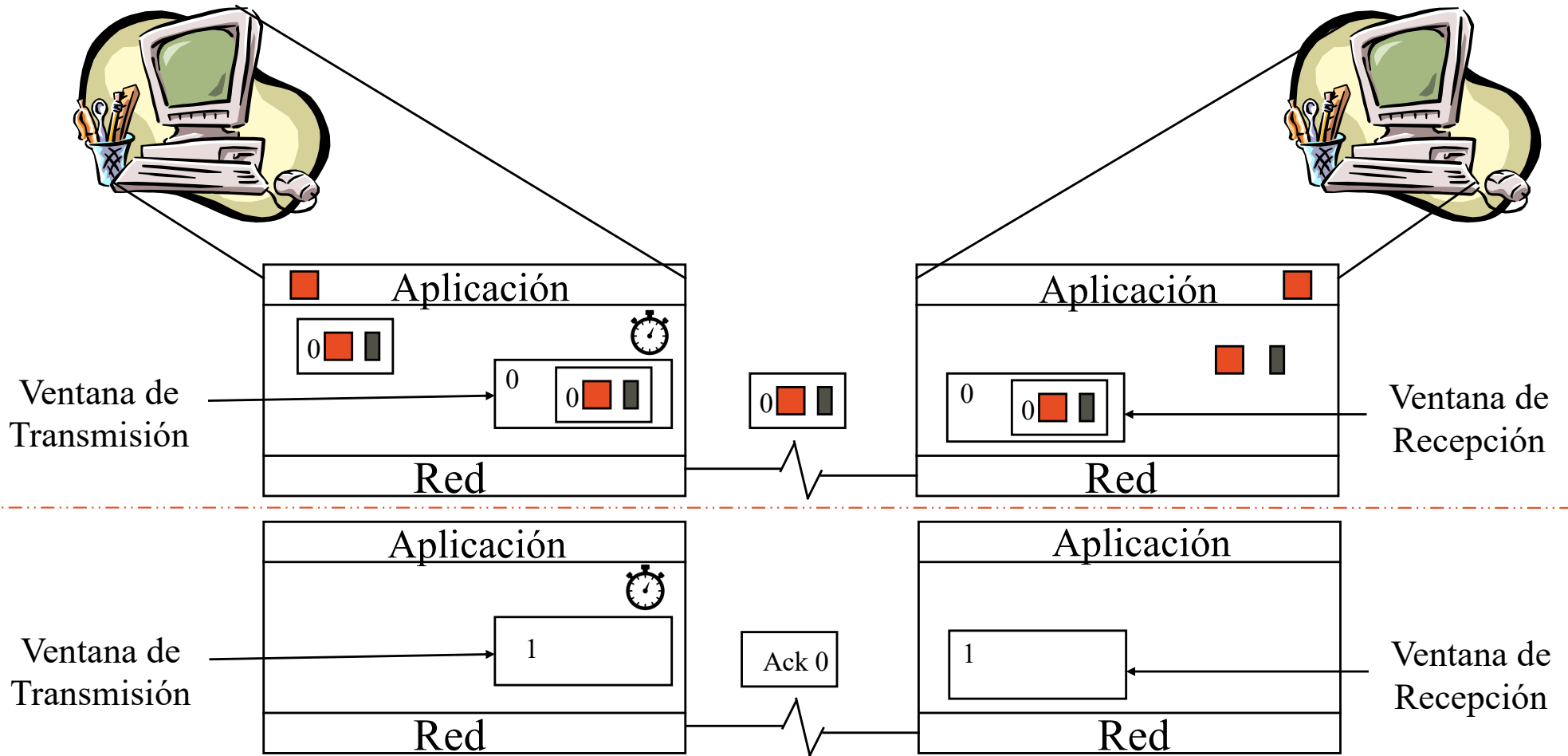


CONTROL DE FLUJO SLIDING WINDOW

- PIPELINING



CONTROL DE FLUJO SLIDING WINDOW



CONTROL DE FLUJO SLIDING WINDOWS



Go back n

- Ventana de recepción de tamaño 1.
- Si se daña un mensaje, descarta los demás de ese punto en adelante
- Números de secuencia =
 $0 - (\text{tamaño de la ventana de transmisión} - 1)$
- Se puede enviar Ack por cada mensaje, pero si se pierde uno de ellos no hay problema porque si llega otro con número de secuencia mayor se supone bien todo de ahí para atrás.

CONTROL DE FLUJO SLIDING WINDOWS

- GO BACK N

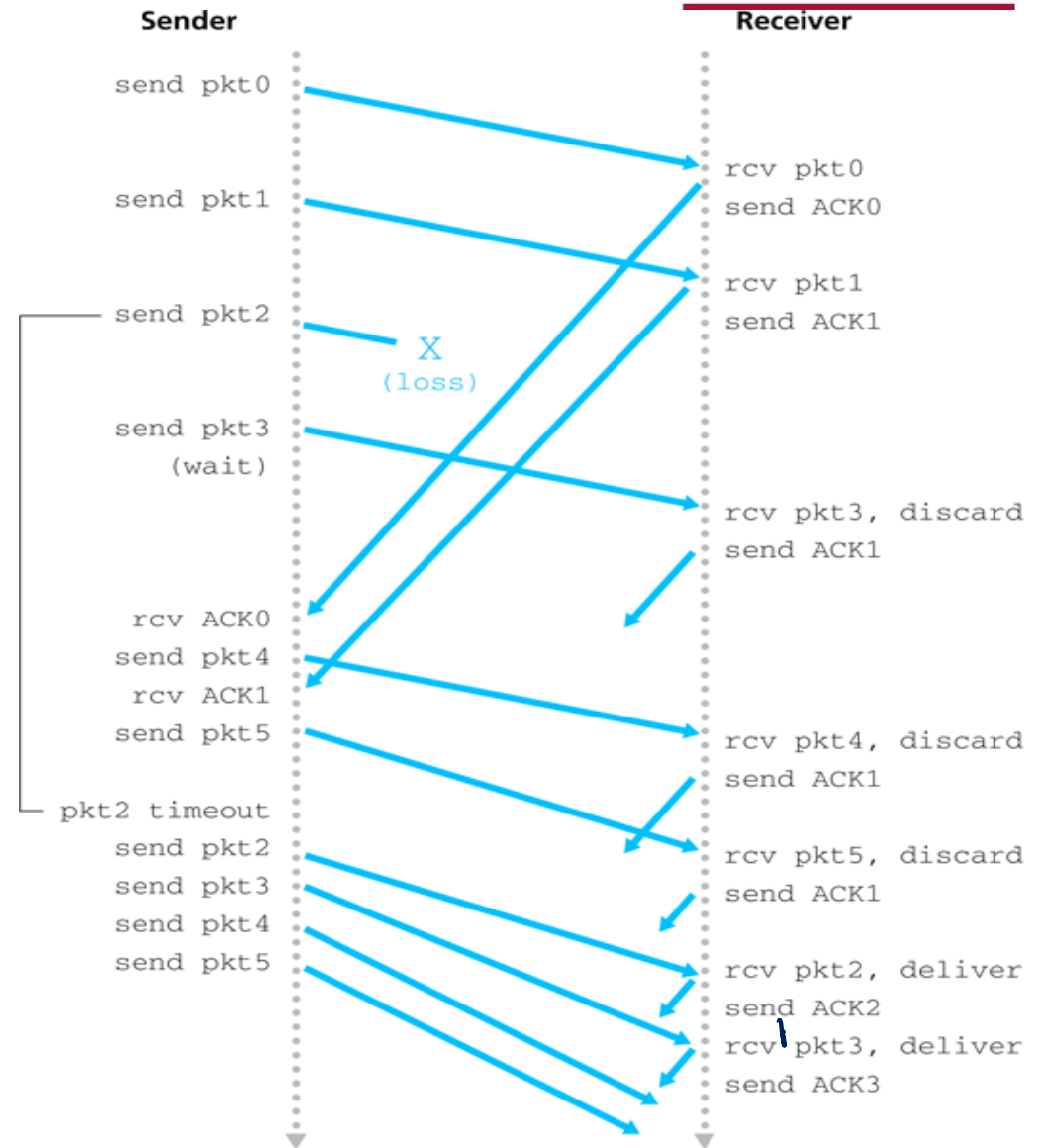


Figure 3.22 ♦ Go-Back-N in operation

CONTROL DE FLUJO SLIDING WINDOW

Go back n

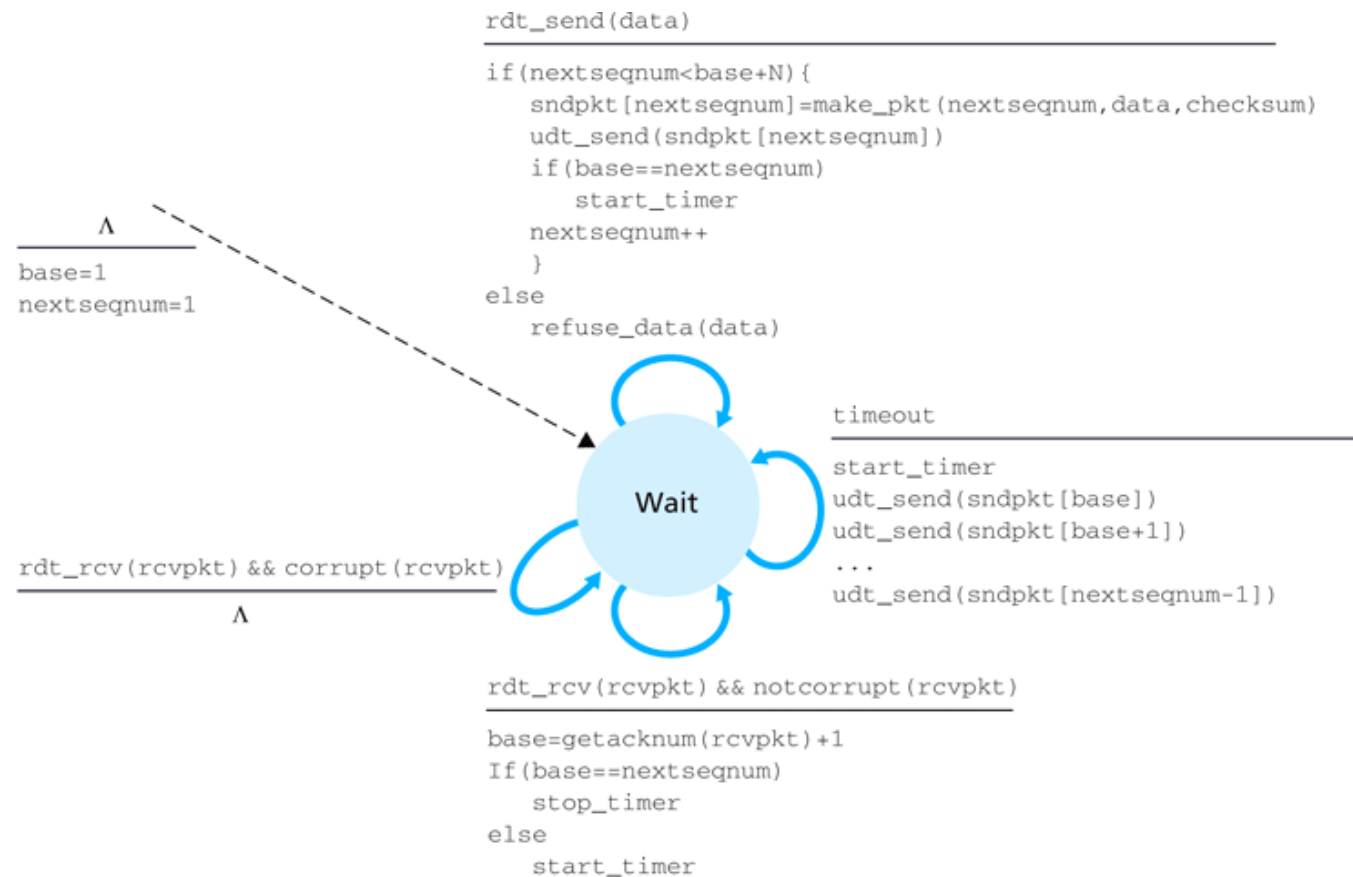


Figure 3.20 ♦ Extended FSM description of GBN sender

CONTROL DE FLUJO SLIDING WINDOW

- GO BACK N

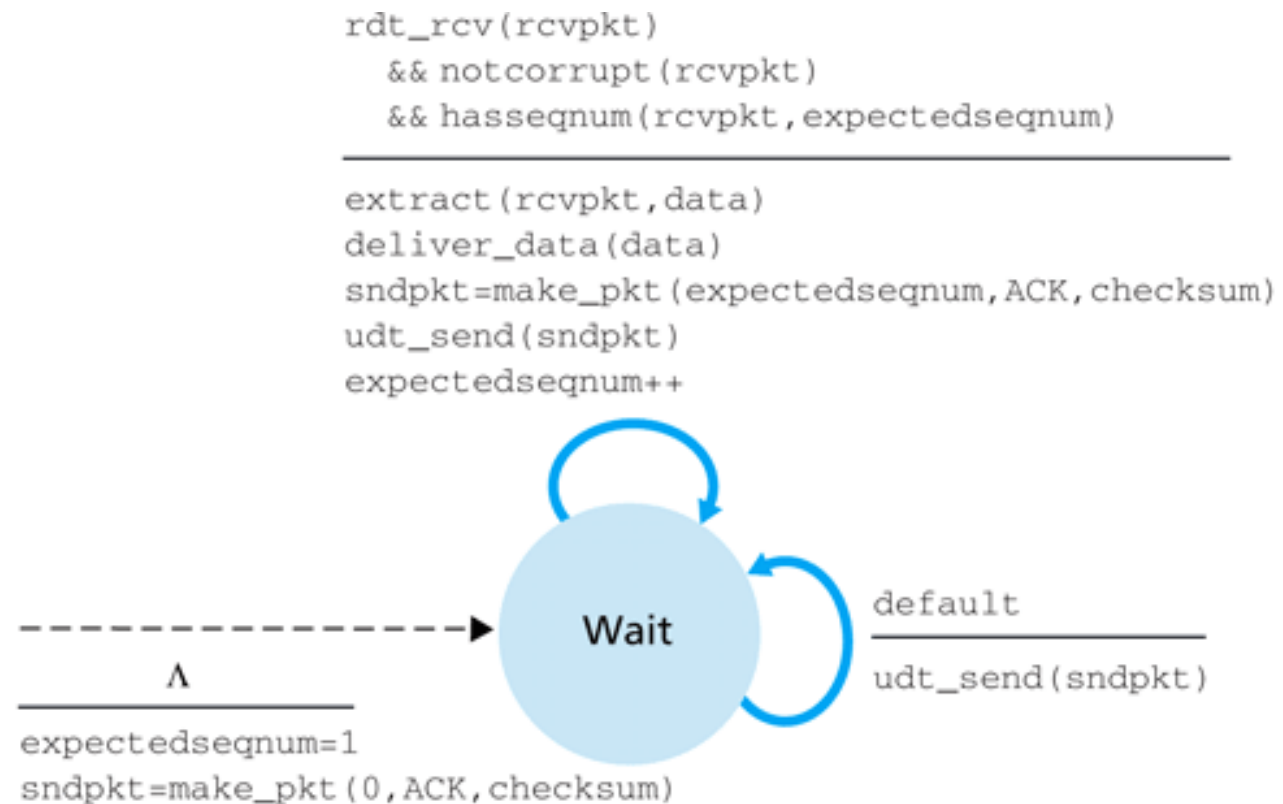


Figure 3.21 ♦ Extended FSM description of GBN receiver

CONTROL DE FLUJO SLIDING WINDOW



Repetición selectiva – selective repeat

- Ventana de recepción de tamaño n .
- Si se daña un mensaje, descarta solo el frame dañado y mantiene en la ventana los paquetes buenos recibidos después del dañado
- Espera recibir retransmisión del dañado y confirma todos los recibidos
- Números de secuencia entre 0 y $(\text{tamaño de la ventana} * 2) - 1$

CONTROL DE FLUJO SLIDING WINDOW

■ REPETICIÓN SELECTIVA

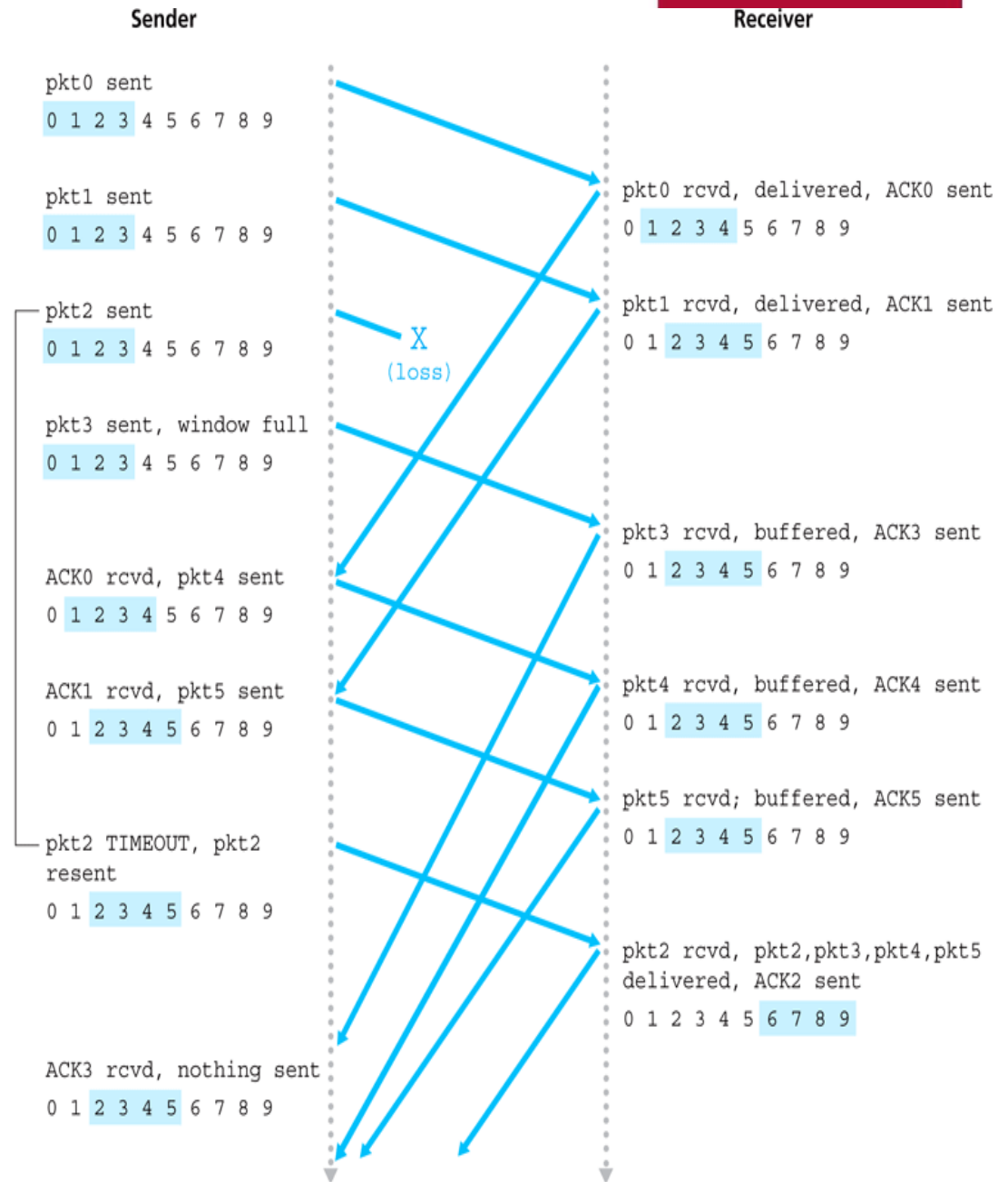


Figure 3.26 ♦ SR operation

CONTROL DE FLUJO SLIDING WINDOW

- REPETICIÓN SELECTIVA — SELECTIVE REPEAT

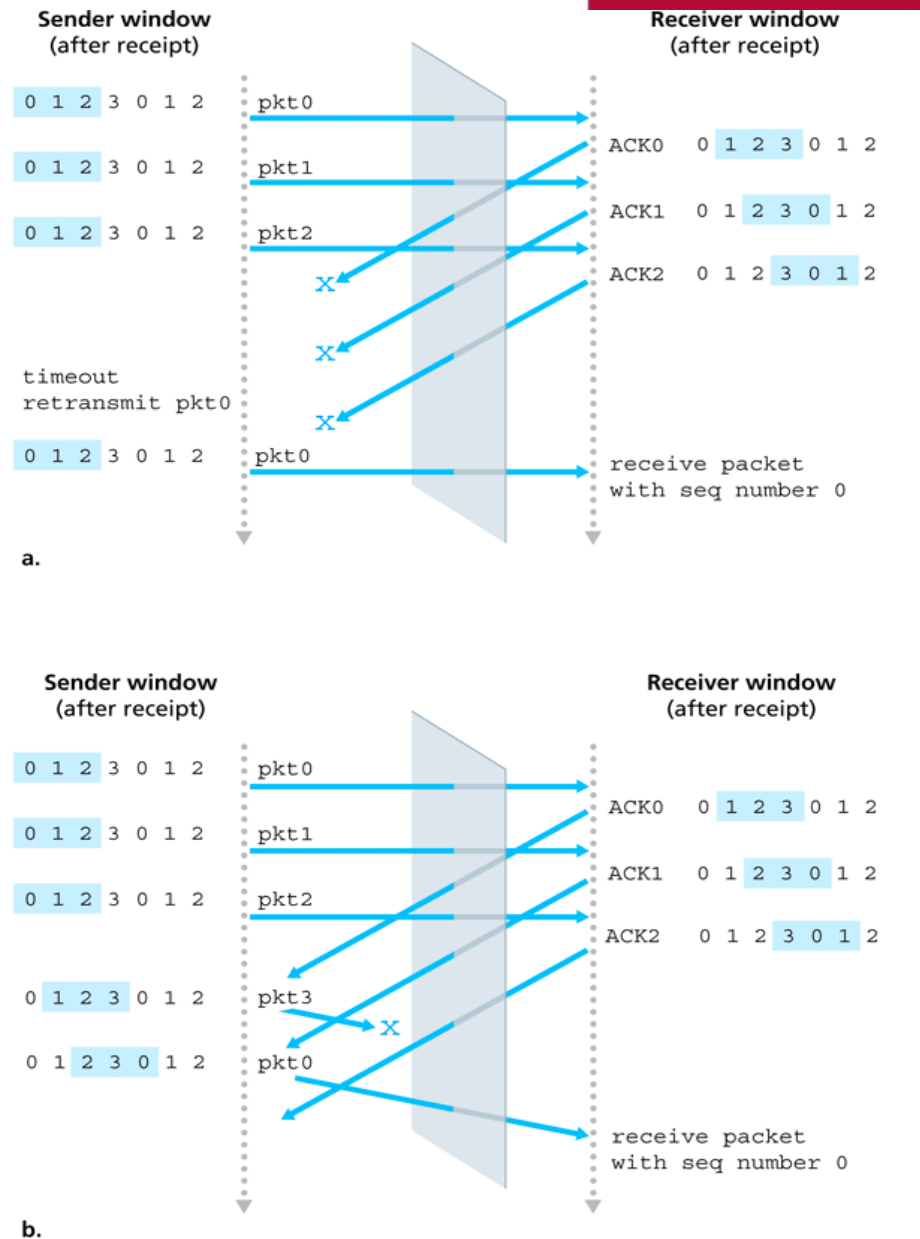


Figure 3.27 ♦ SR receiver dilemma with too-large windows: A new packet or a retransmission?

Computer Networking: A Top-Down Approach, 4/E. James F. Kurose, Keith W. Ross,

CONTROL DE FLUJO SLIDING WINDOW

- REPETICIÓN SELECTIVA —
SELECTIVE REPEAT

Mechanism	Use, Comments
Checksum	Used to detect bit errors in a transmitted packet.
Timer	Used to timeout/retransmit a packet, possibly because the packet (or its ACK) was lost within the channel. Because timeouts can occur when a packet is delayed but not lost (premature timeout), or when a packet has been received by the receiver but the receiver-to-sender ACK has been lost, duplicate copies of a packet may be received by a receiver.
Sequence number	Used for sequential numbering of packets of data flowing from sender to receiver. Gaps in the sequence numbers of received packets allow the receiver to detect a lost packet. Packets with duplicate sequence numbers allow the receiver to detect duplicate copies of a packet.
Acknowledgment	Used by the receiver to tell the sender that a packet or set of packets has been received correctly. Acknowledgments will typically carry the sequence number of the packet or packets being acknowledged. Acknowledgments may be individual or cumulative, depending on the protocol.
Negative acknowledgment	Used by the receiver to tell the sender that a packet has not been received correctly. Negative acknowledgments will typically carry the sequence number of the packet that was not received correctly.
Window, pipelining	The sender may be restricted to sending only packets with sequence numbers that fall within a given range. By allowing multiple packets to be transmitted but not yet acknowledged, sender utilization can be increased over a stop-and-wait mode of operation. We'll see shortly that the window size may be set on the basis of the receiver's ability to receive and buffer messages, or the level of congestion in the network, or both.

Table 3.1 ♦ Summary of reliable data transfer mechanisms and their use

TRANSMISIÓN FIABLE DE DATOS EN TCP

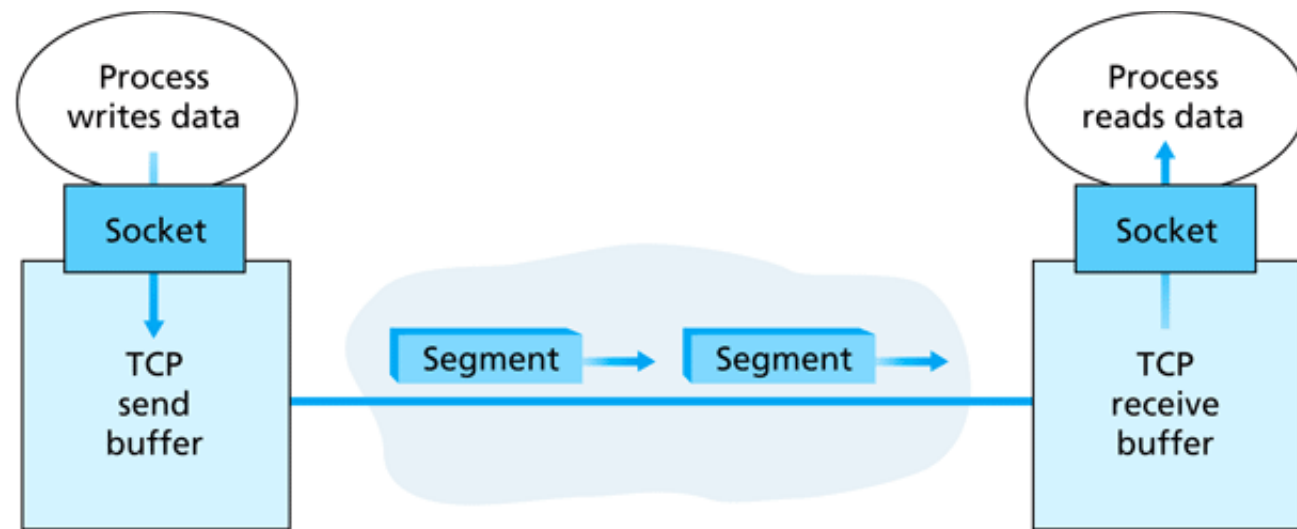


Figure 3.28 ♦ TCP send and receive buffers

TRANSMISIÓN FIABLE DE DATOS EN TCP

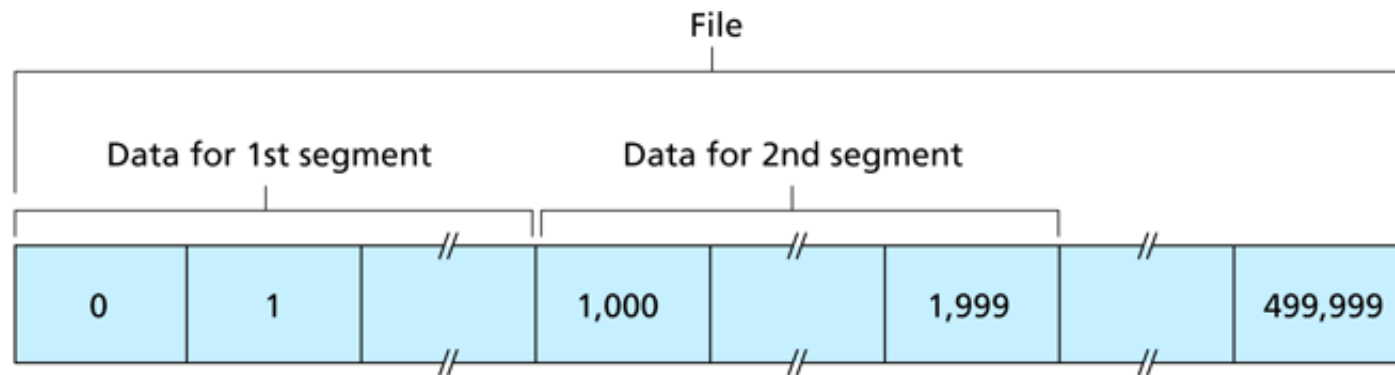


Figure 3.30 ♦ Dividing file data into TCP segments

Computer Networking: A Top-Down Approach, 4/E. **James F. Kurose, Keith W. Ross,**

Número de secuencia

Número de confirmación (en bytes)

Timeouts

TRANSMISIÓN FIABLE DE DATOS EN TCP

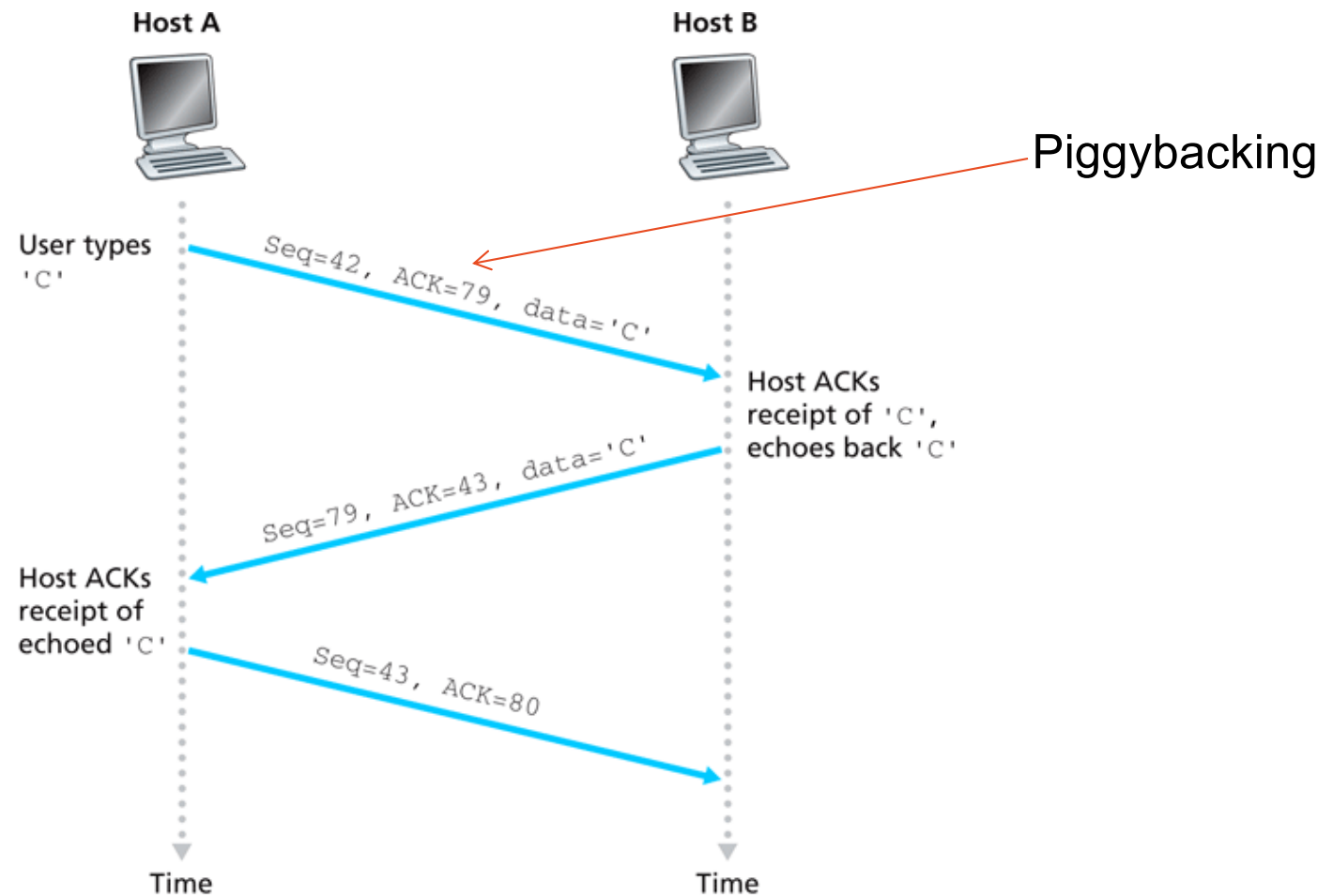


Figure 3.31 ♦ Sequence and acknowledgement numbers for a simple Telnet application over TCP



OTRO EJEMPLO



TRANSMISIÓN FIABLE DE DATOS EN TCP

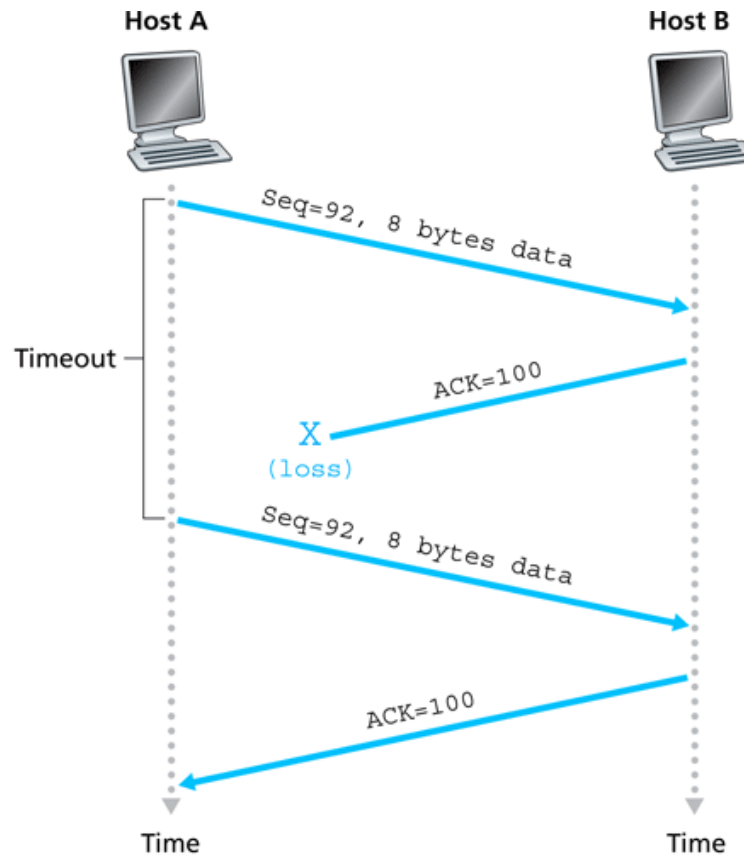


Figure 3.34 ♦ Retransmission due to a lost acknowledgment

TRANSMISIÓN FIABLE DE DATOS EN TCP

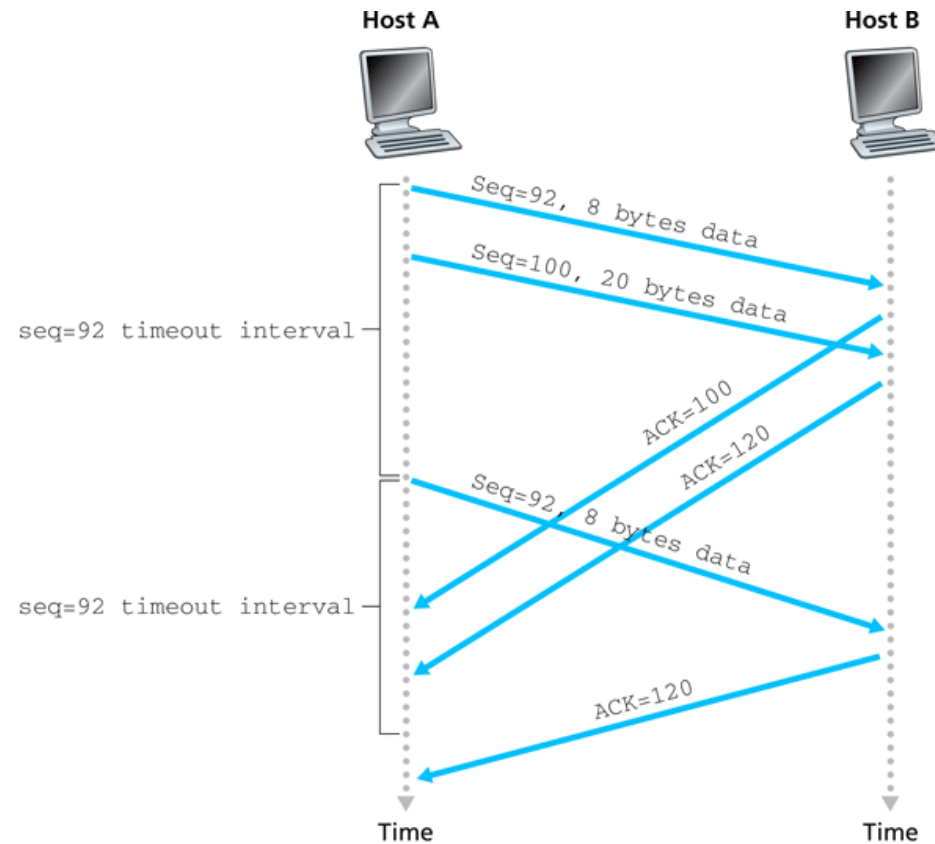


Figure 3.35 ♦ Segment 100 not retransmitted

TRANSMISIÓN FIABLE DE DATOS EN TCP

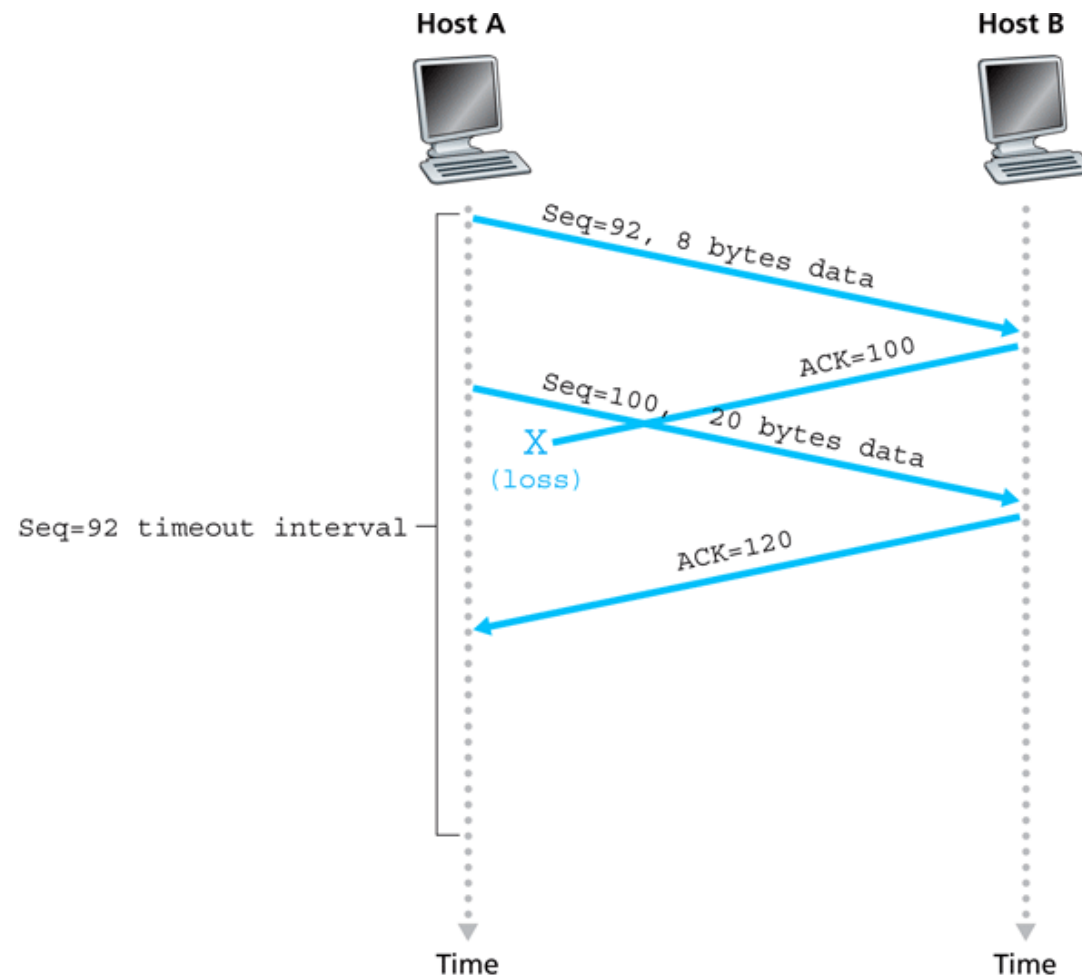


Figure 3.36 ♦ A cumulative acknowledgment avoids retransmission of the first segment.

TRANSMISIÓN FIABLE DE DATOS EN TCP

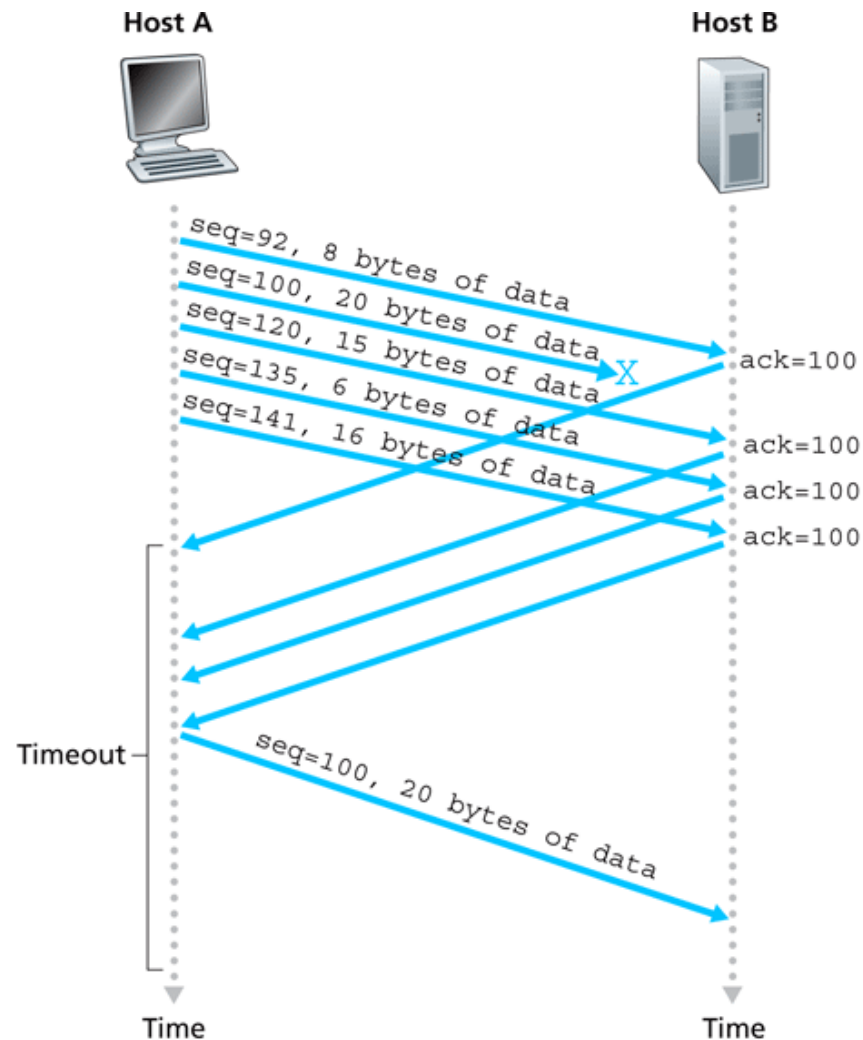
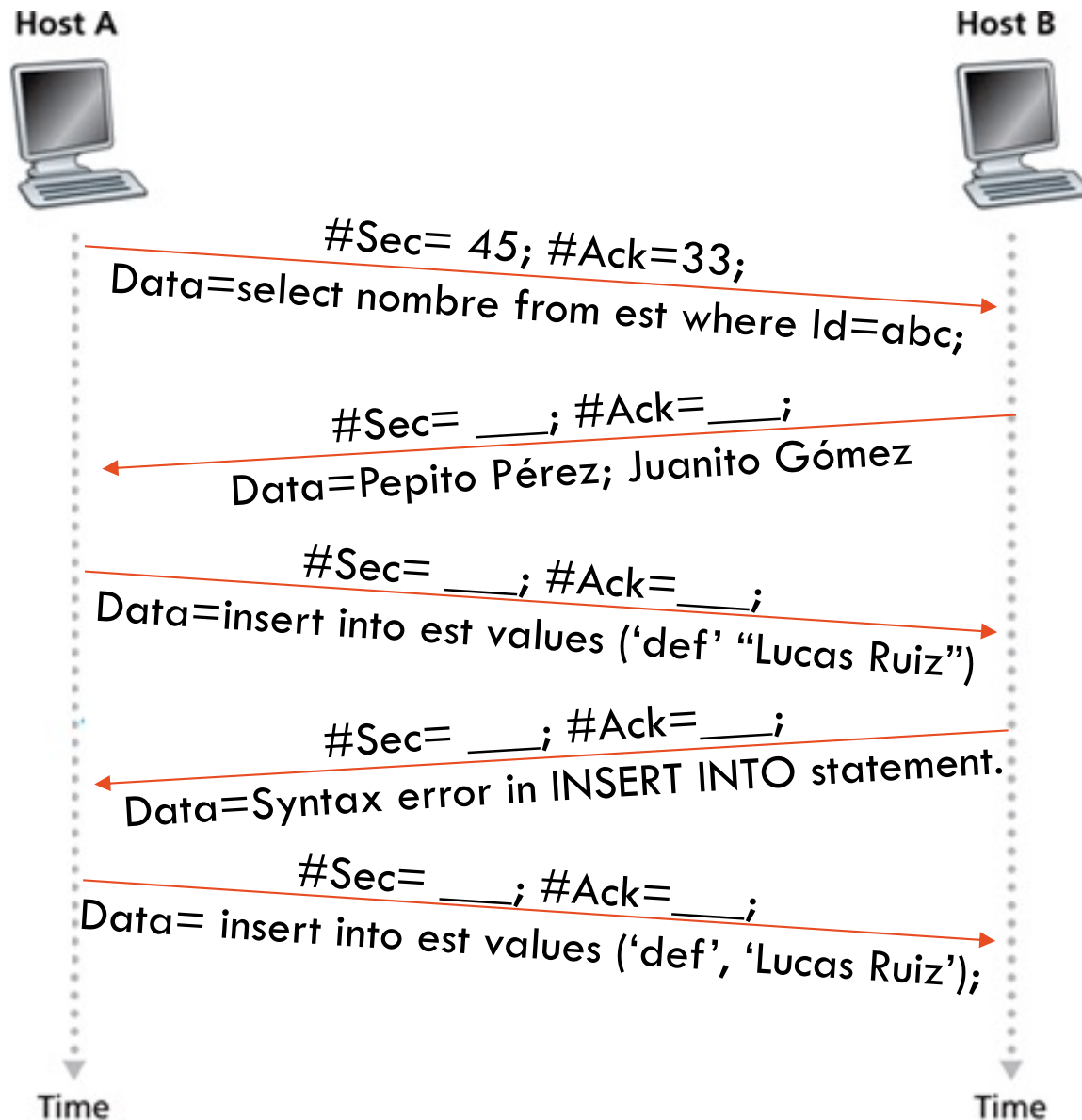
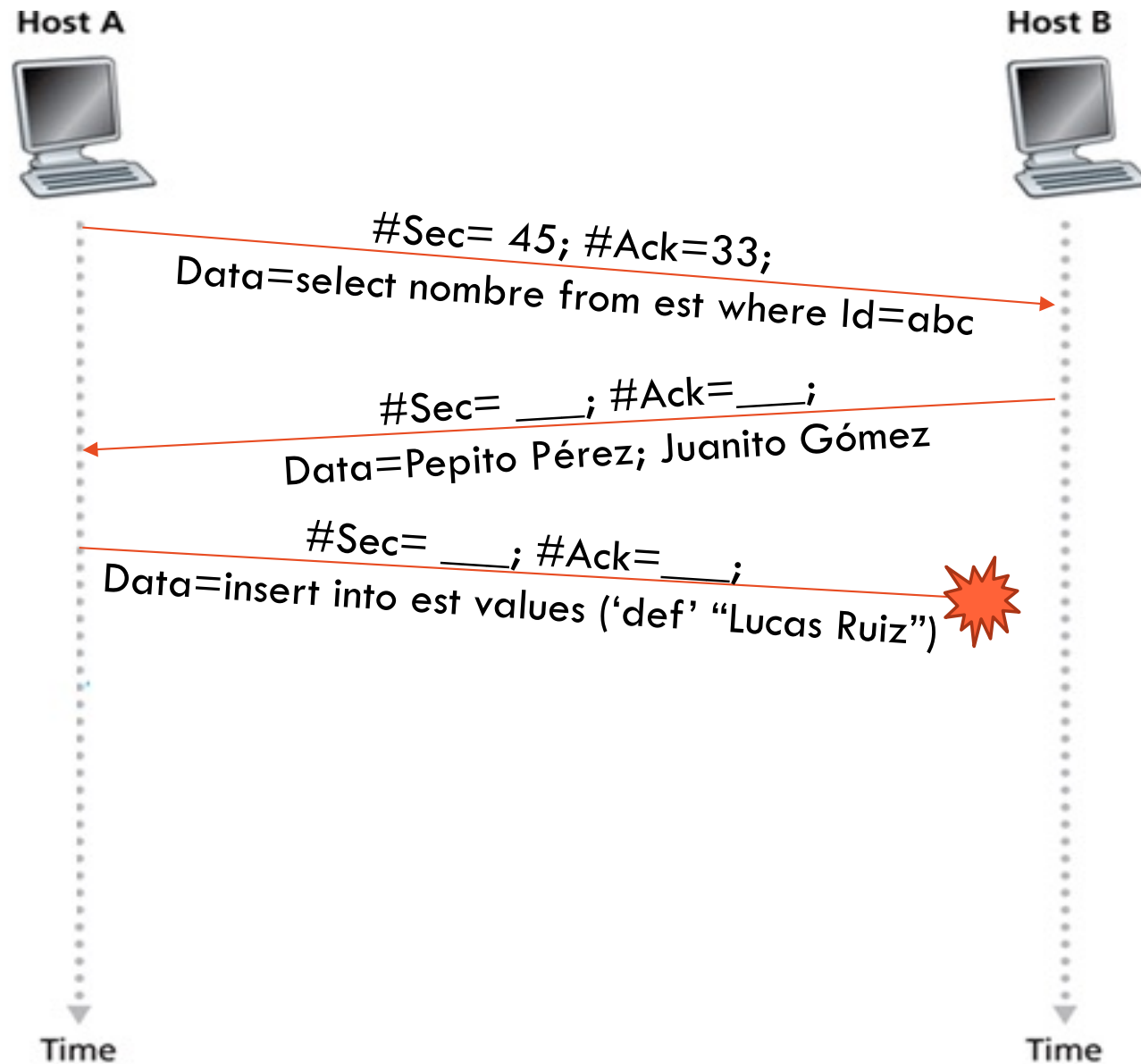


Figure 3.37 ♦ Fast retransmit: retransmitting the missing segment before the segment's timer expires.

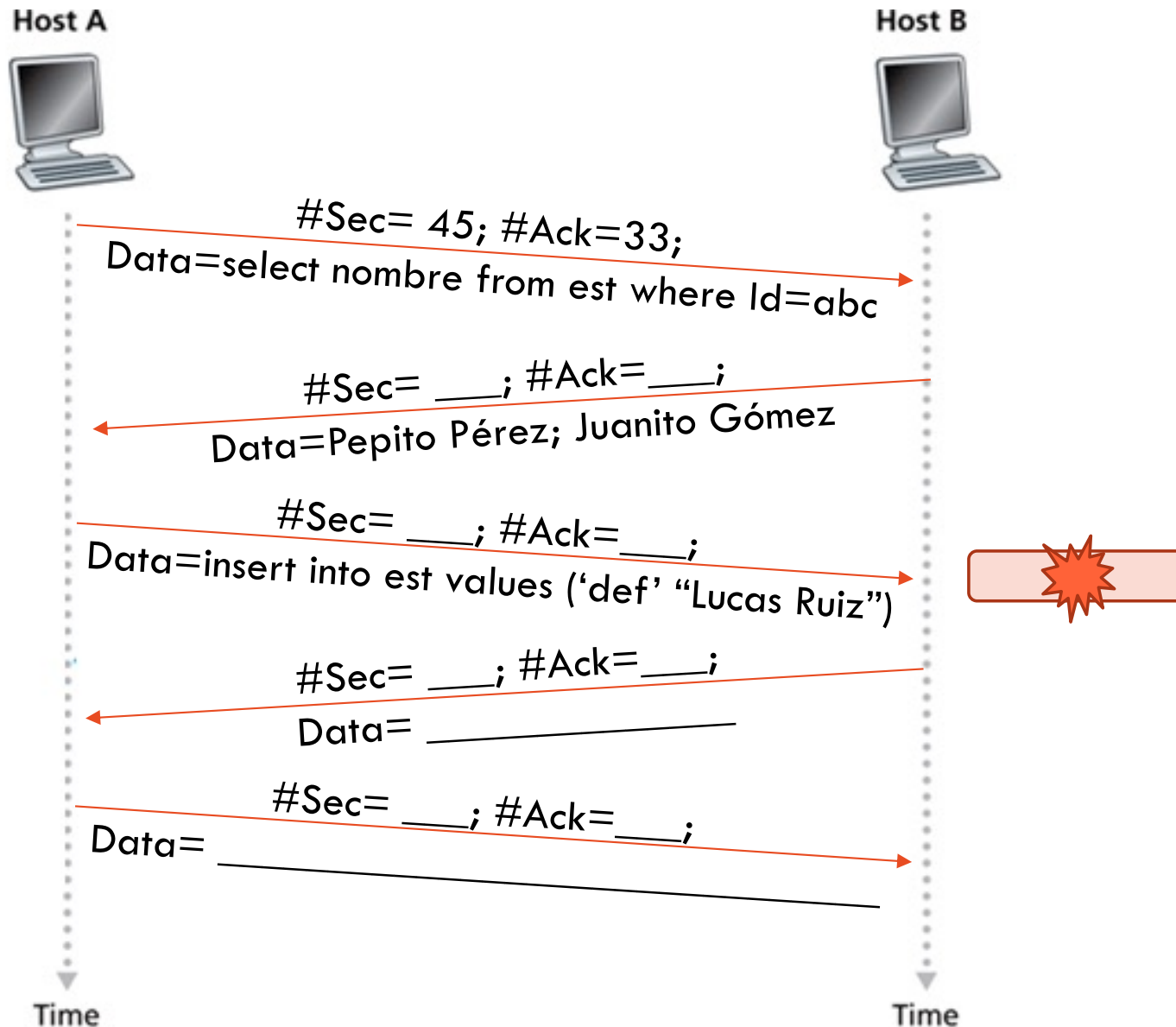
UN EJERCICIO MÁS



UN EJERCICIO MÁS



UN EJERCICIO MÁS



TRANSMISIÓN FIABLE DE DATOS EN TCP

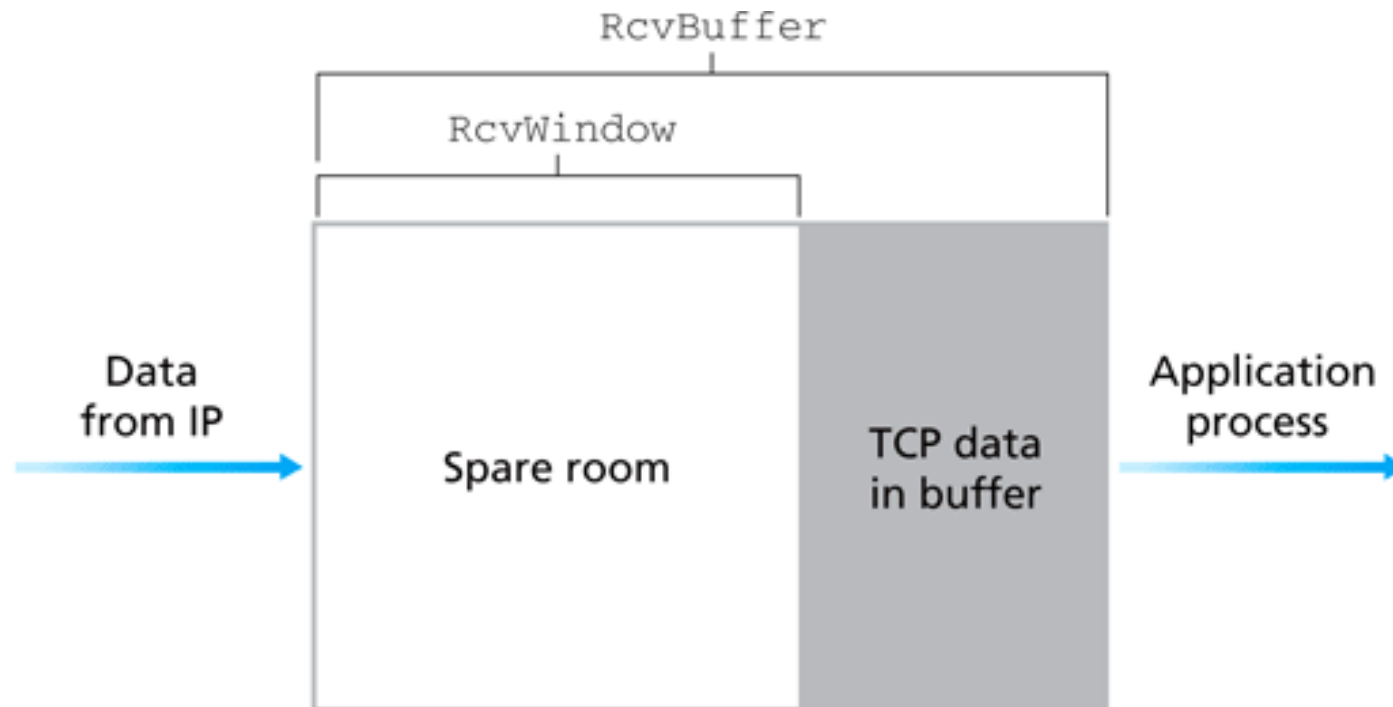


Figure 3.38 ♦ The receive window (RcvWindow) and the receive buffer (RcvBuffer)



ADMINISTRACIÓN DE LA CONEXIÓN

OAC

Proceso

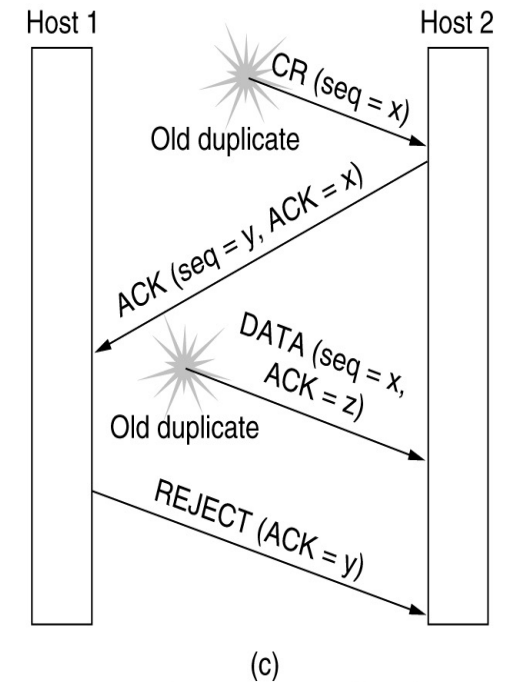
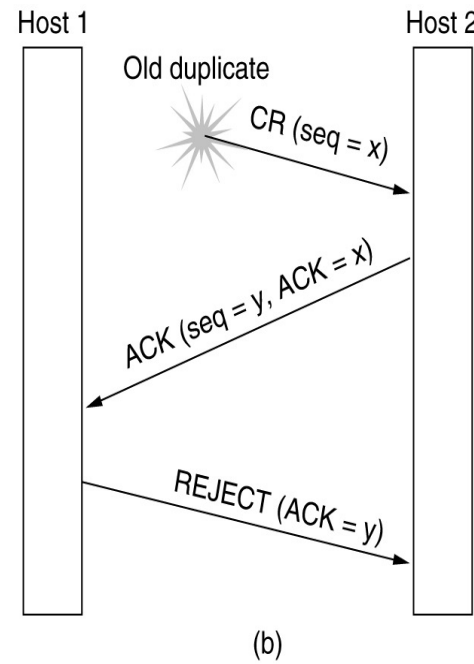
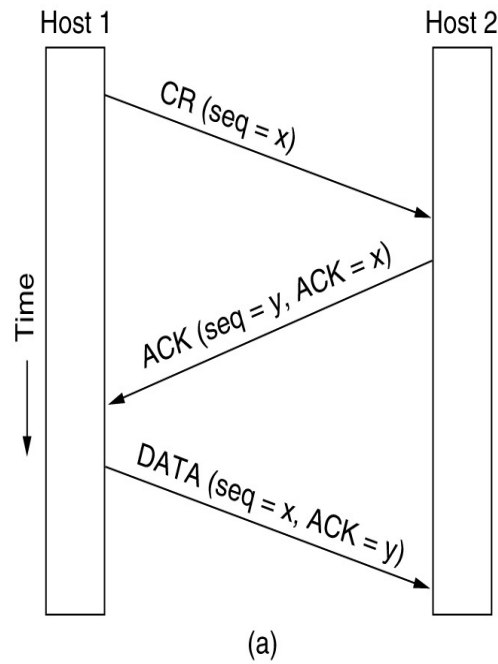
- Conexión
- Transmisión
- Desconexión

Recuperación de la conexión

Algoritmo three-way-handshake

ADMINISTRACIÓN DE LA CONEXIÓN TCP

Conexión



ADMINISTRACIÓN DE LA CONEXIÓN TCP

Conexión

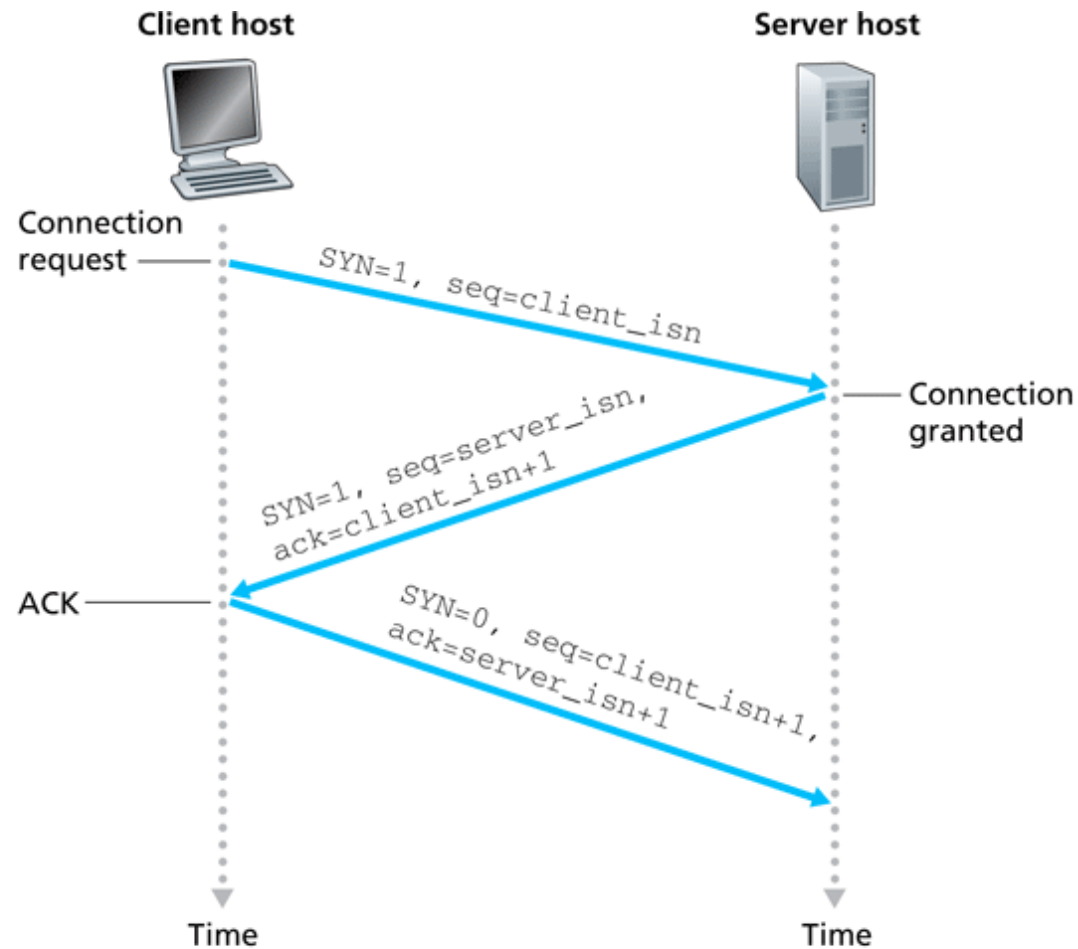
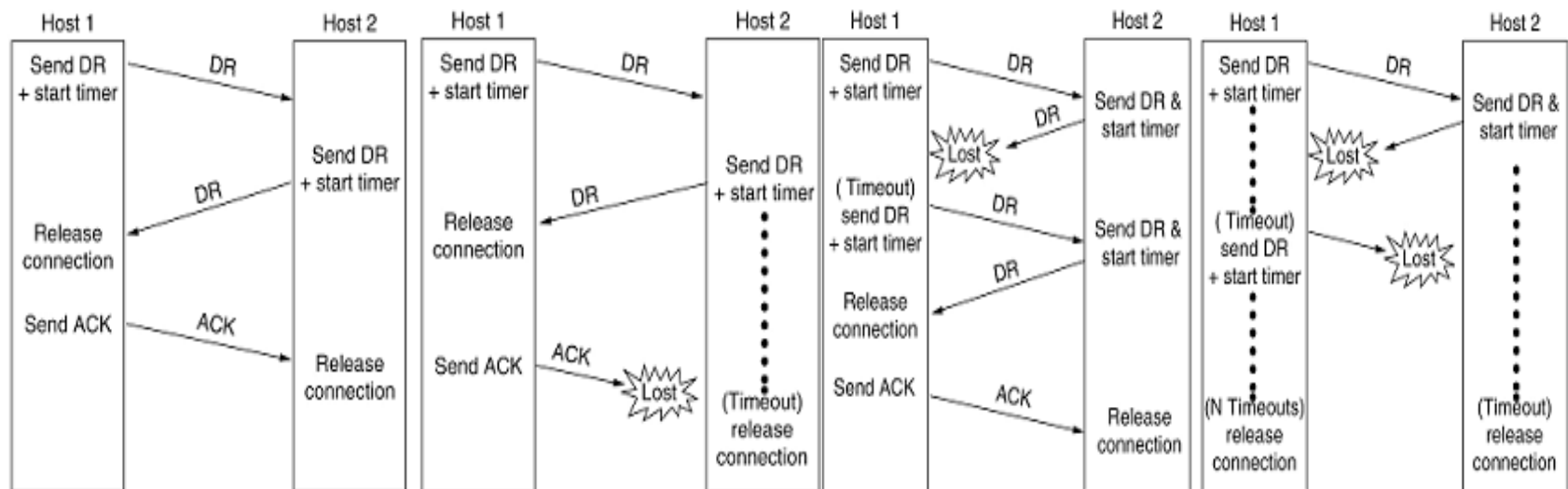


Figure 3.39 ♦ TCP three-way handshake: segment exchange

ADMINISTRACIÓN DE LA CONEXIÓN TCP

Desconexión



ADMINISTRACIÓN DE LA CONEXIÓN TCP

Desconexión

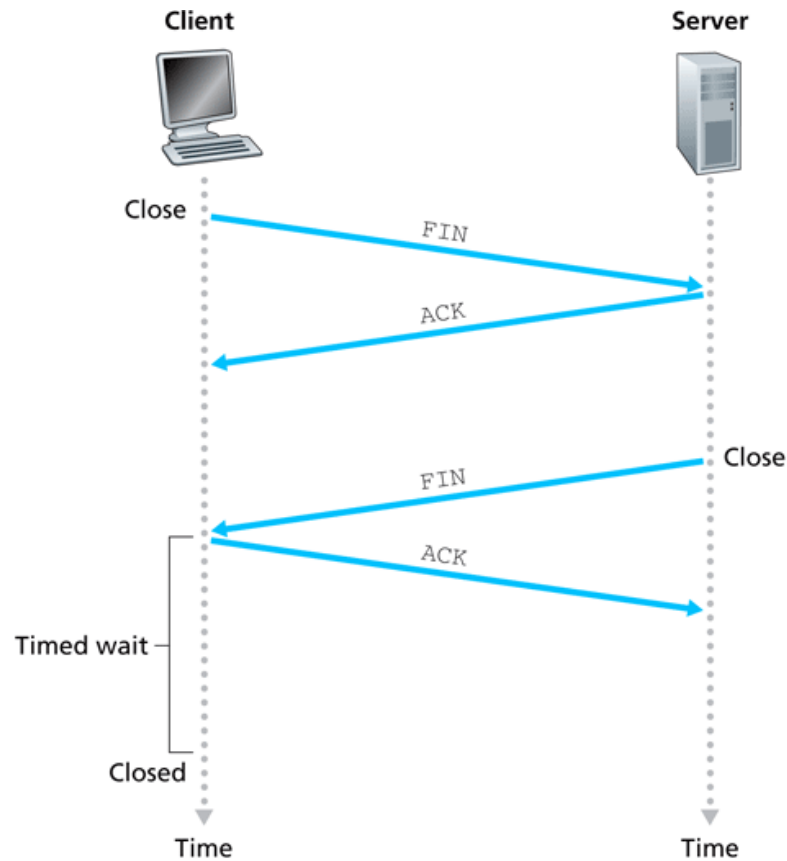


Figure 3.40 ♦ Closing a TCP connection

ADMINISTRACIÓN DE LA CONEXIÓN TCP

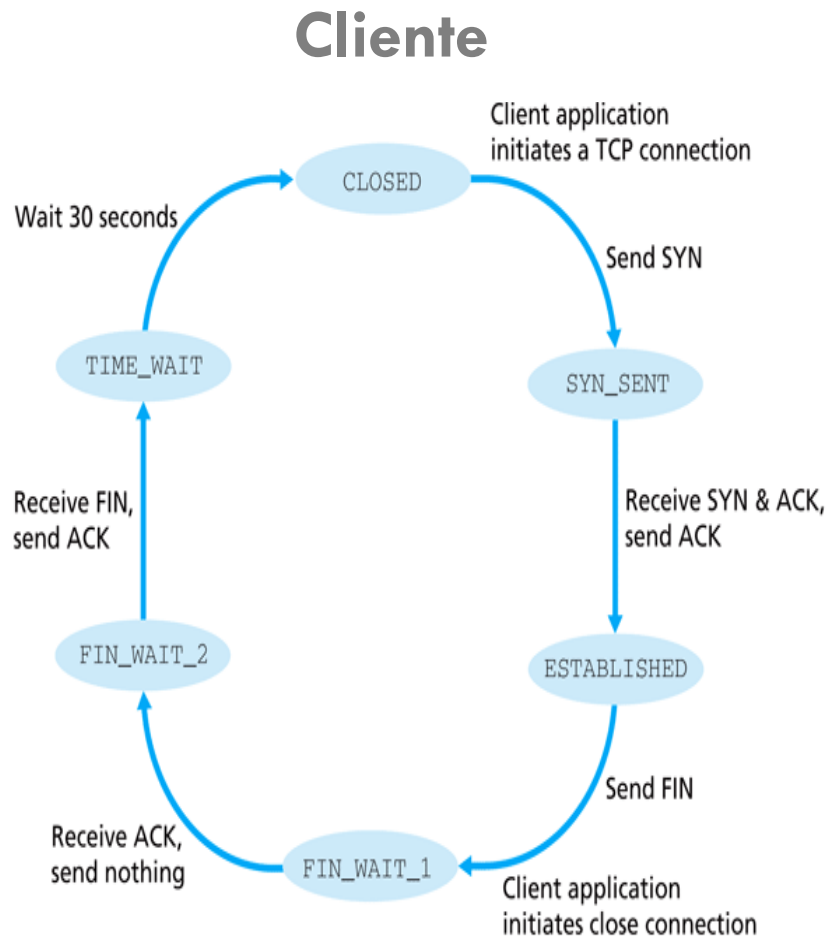


Figure 3.41 ♦ A typical sequence of TCP states visited by a client TCP

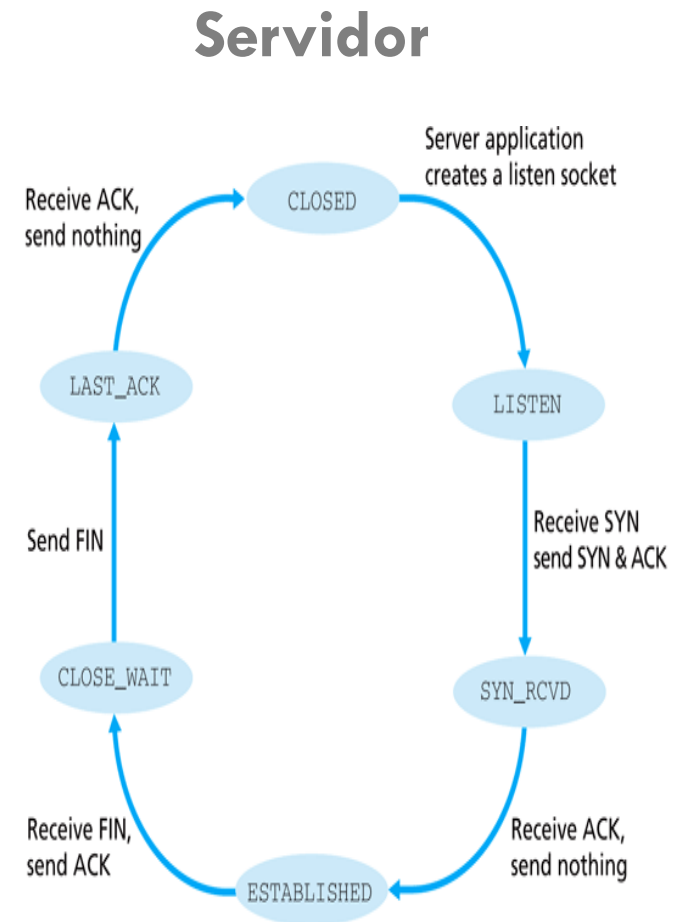
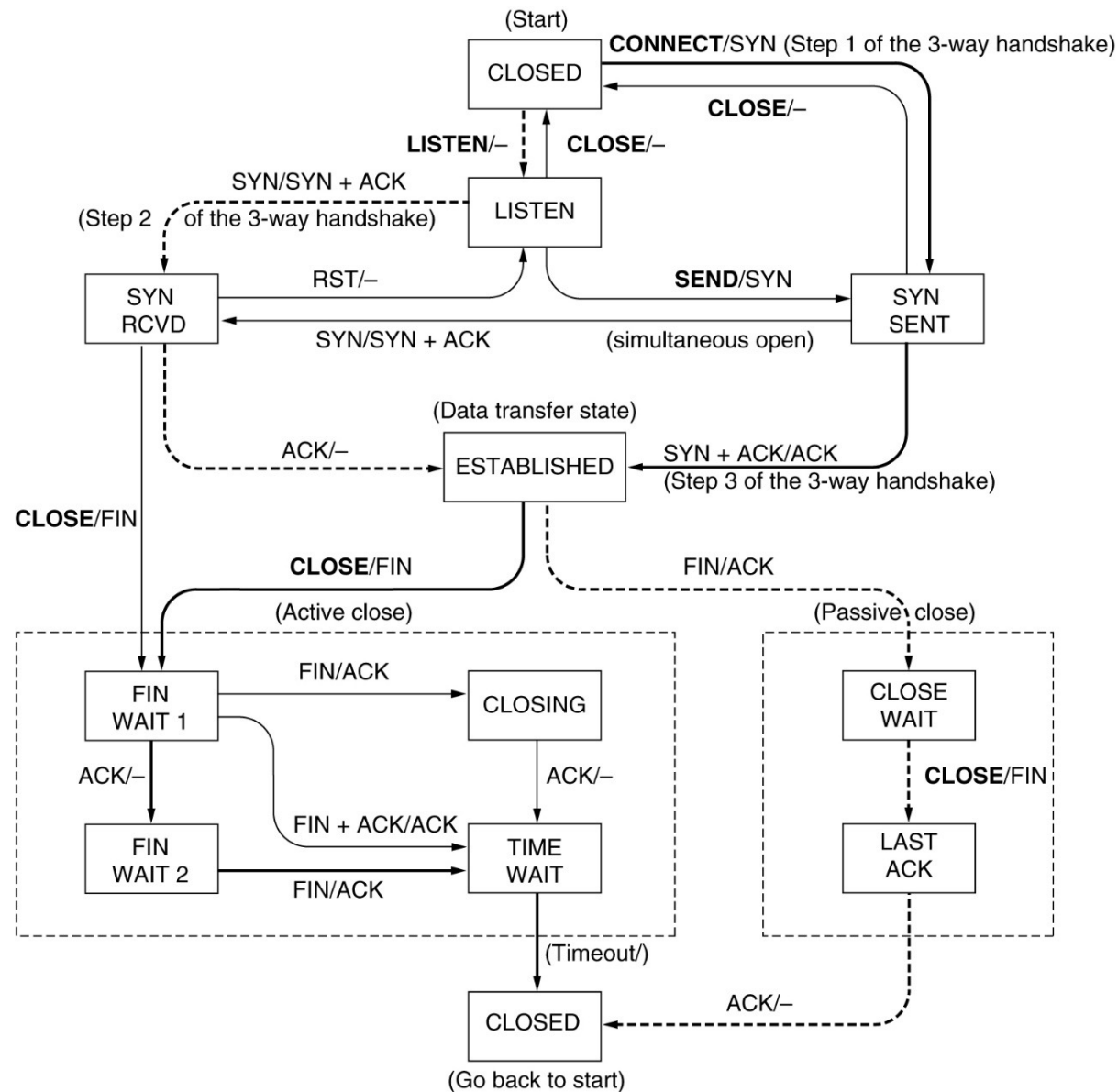


Figure 3.42 ♦ A typical sequence of TCP states visited by a server-side TCP

ADMINISTRACIÓN DE LA CONEXIÓN TCP



ENCABEZADO TCP

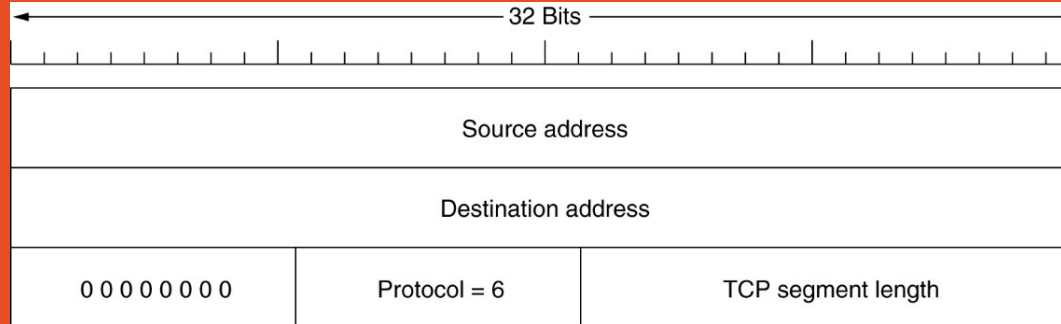
Puerto desde
donde se
origina el
mensaje

Número de Confirmación:
Identifica el siguiente byte

Puerto a donde
debe llegar el
mensaje

Apuntador a datos

Para control de errores. Incluye: Datos +
encabezado + pseudoencabezado



Tam
encab
palabra

Se usa
opciones
en el en
ejempl

máximo de mens
recibe.

Bandera
Establecir
sincroniza
de núm

(SYN, ACK, FIN)

finaliza
que el e
tiene m
para tr

Indica espacio libre
que tiene para recibir
dentro de la ventana de
recepción. En bytes

Indica la
posición de los datos del
mensaje dentro del flujo
total de datos a
transmitir

CONTROL DE CONGESTIÓN

Por qué se presenta?

- Muchos paquetes para una misma salida
- Enrutadores con procesador lento o poca memoria
- Poco AdeB

Algoritmos

- Chocket Packets (Paq a origen para que disminuya la tasa de envío)
- Chocket Packets hop by hop
- Load shedding (el más viejo o el más nuevo)
- Jitter (audio – video. Adelanta - atrasa)

CONTROL DE CONGESTIÓN EN TCP

Ventanas de control de flujo

- Ventana de transmisión
- Ventana de Recepción

Ventana de congestión

- Inicialmente: Tamaño del segmento máximo establecido por la conexión
- Si llega bien, duplica. Sino, se va a la mitad

Algoritmo de arranque lento

- Añade un Threshold - umbral

CONTROL DE CONGESTIÓN EN TCP

ALGORITMO DE ARRANQUE LENTO

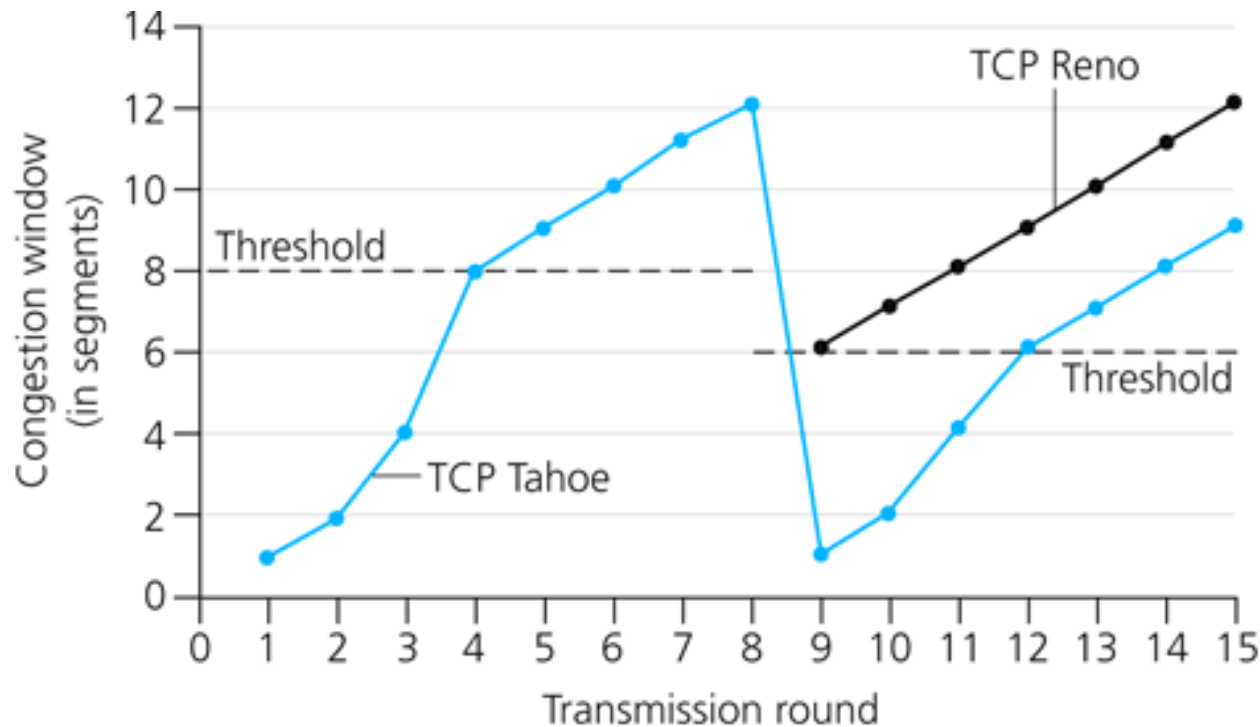


Figure 3.53 ♦ Evolution of TCP's congestion window (Tahoe and Reno)

CONTROL DE CONGESTIÓN EN TCP

ALGORITMO DE ARRANQUE LENTO

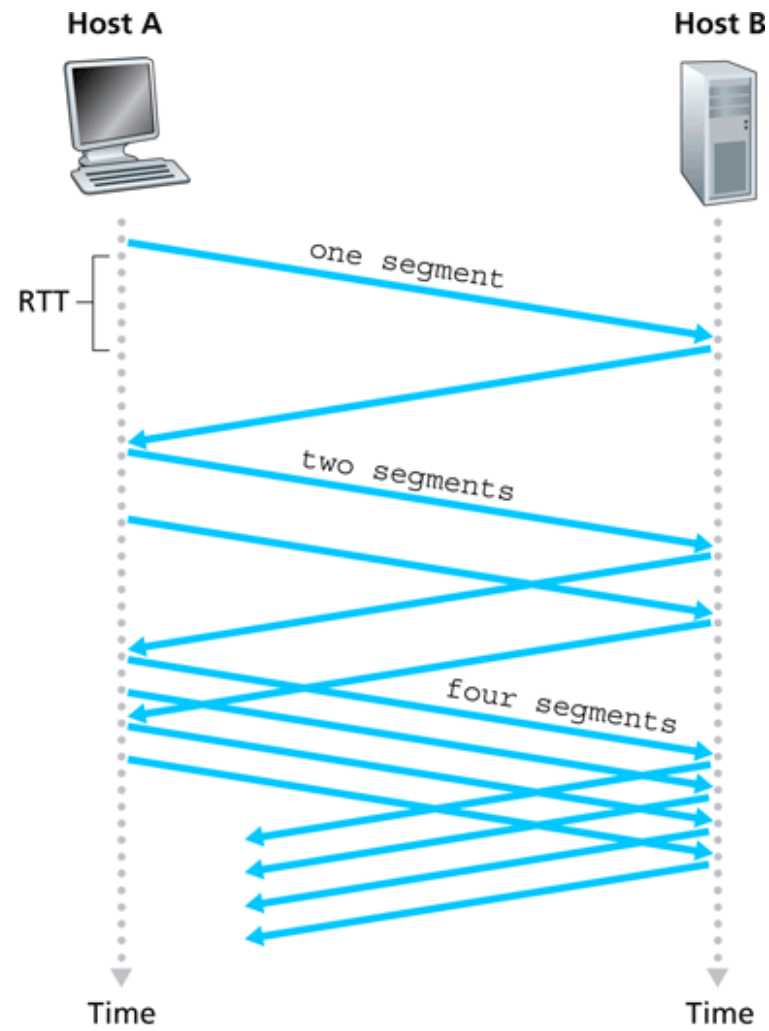


Figure 3.52 ♦ TCP slow start

TEMPORIZADORES EN TCP

Temporizador de persistencia

- Evita el interbloqueo
- Cuando se pierden “informes” del tamaño de la ventana de recepción libre (en un momento 0 y luego libre).

Temporizador de seguir con vida - keepalive timer

- Cuando no se han transmitido datos durante “mucho” tiempo

Timed wait

- Cuando termina una conexión
- Es del doble del máximo tiempo de vida de un paquete
- Para asegurar que no queden en la red paquetes de la conexión

Temporizador de retransmisión



CONTROL DE FLUJO

- Se usa para que transmisores rápidos no ahoguen a receptores lentos
- Generalmente los protocolos tiene implementadas una de las siguientes maneras para hacer este control de flujo
 - Un momento en el que negocian la velocidad
 - El receptor va indicando al transmisor cuando puede enviar mas mensajes.

Mecanismos:

- Simplex
 - Unrestricted (Canal perfecto)
 - For a noisy channel - Stop and wait
 - Pueden dañarse datos
 - Pueden perderse datos
- Sliding Window
 - Go back n
 - Selective repeat



CONTROL DE FLUJO SIMPLEX

Sobre un canal perfecto

- Mas sencillo
- Transmisión de datos en una sola dirección
- No hay problema en el receptor para recibir datos. Buffers ilimitados.
- Nunca se pierden ni dañan datos

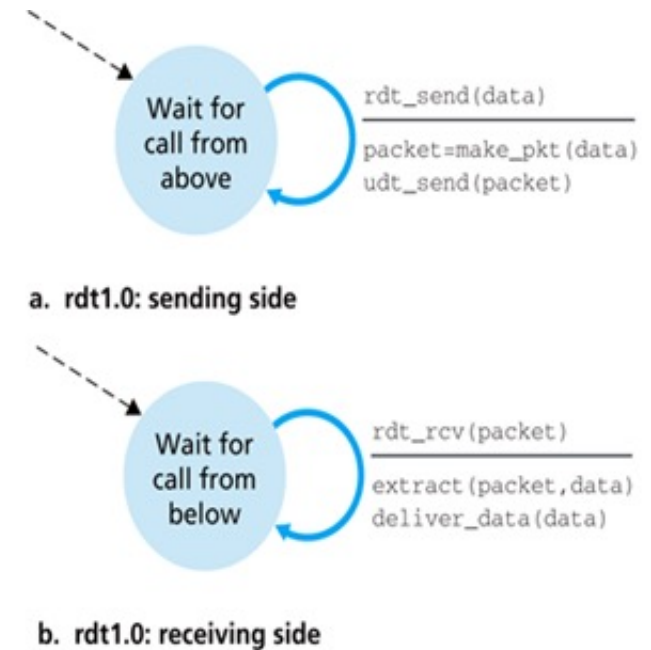


Figure 3.9 ♦ rdt1.0 – A protocol for a completely reliable channel

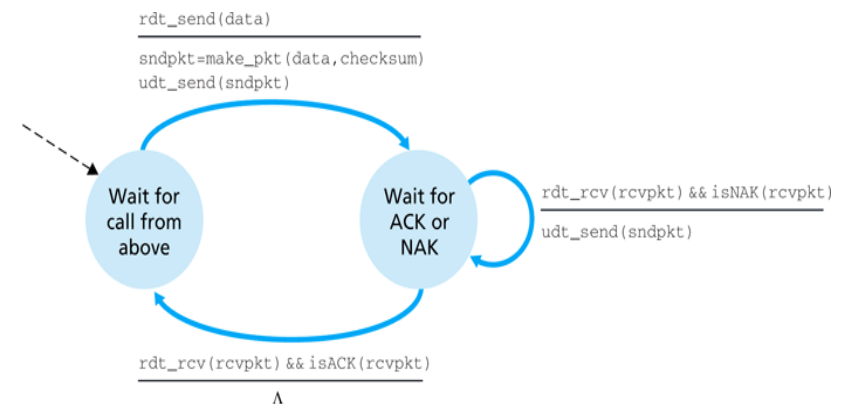
CONTROL DE FLUJO SIMPLEX

Sobre un canal ruidoso – Stop and wait (pueden dañarse datos)

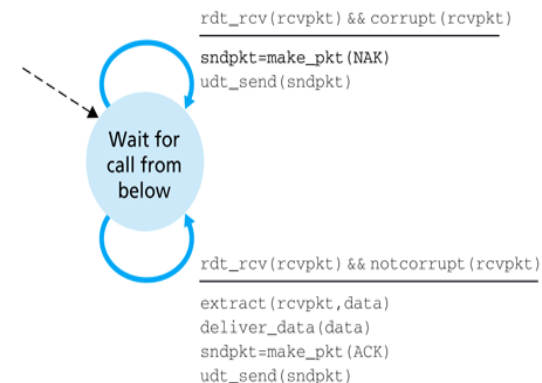
- Parada y espera



- Se pueden presentar errores durante la transmisión
- El receptor
 - Detección de errores: verifica que los datos que recibe se encuentren sin errores antes de pasarlos a la capa superior
 - Feedback: Envía mensaje a origen indicando el estado de los datos (ACK o NAK)
- El origen retransmite si se requiere



a. rdt2.0: sending side



b. rdt2.0: receiving side

Figure 3.10 ♦ rdt2.0—A protocol for a channel with bit errors

CONTROL DE FLUJO SIMPLEX

Sobre un canal ruidoso (pueden dañarse datos)

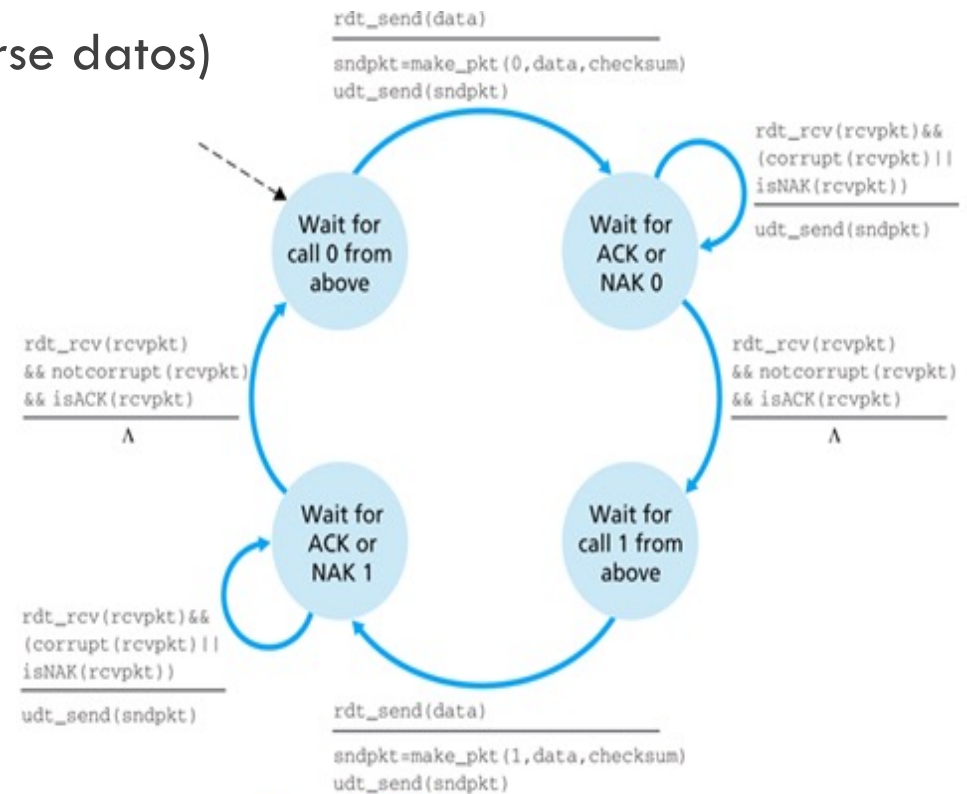


Figure 3.11 ♦ rdt2.1 sender

CONTROL DE FLUJO SIMPLEX

Sobre un canal ruidoso (pueden dañarse datos)

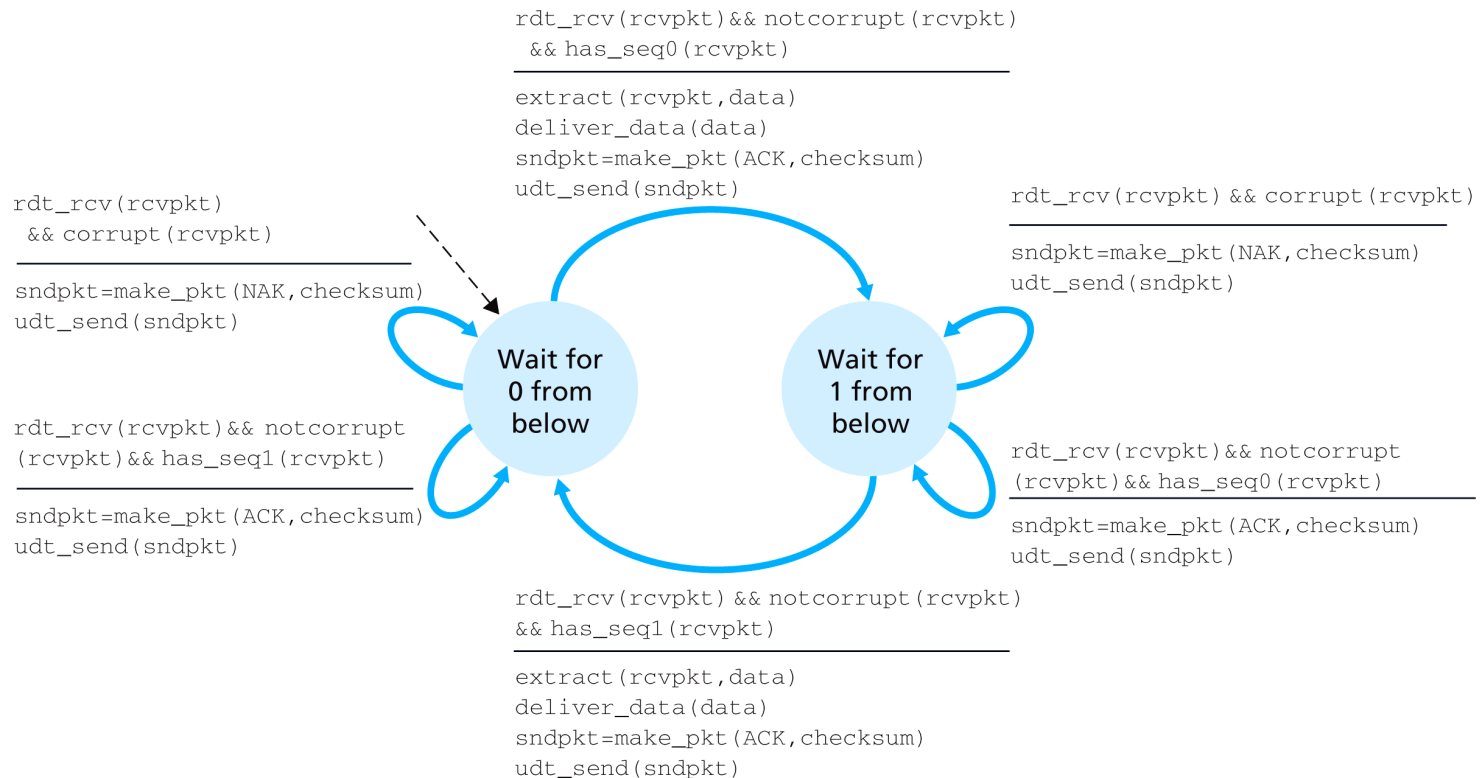


Figure 3.12 ♦ rdt2.1 receiver

CONTROL DE FLUJO SIMPLEX

SOBRE UN CANAL RUIDOSO (PUEDEN PERDERSE DATOS)



✦ Problemas:

✦ ¿Qué pasa si se pierde un mensaje?

✦ ¿Qué pasa si lo que se daña un mensaje de confirmación de recibo (ACK o NAK)?

✦ ¿Qué pasa si se pierde una confirmación de recibo?

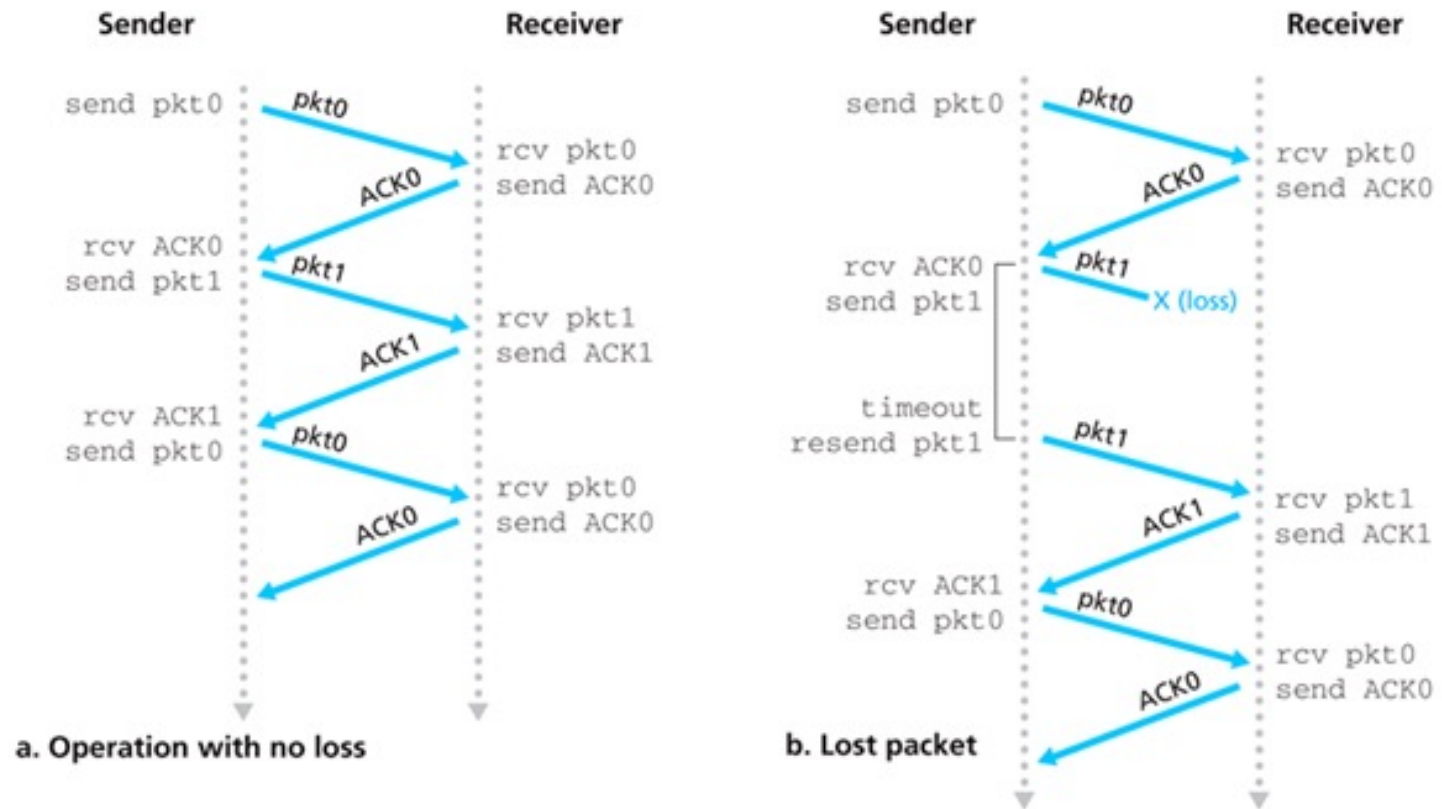
✦ Soluciones

✦ Números de secuencia(1 /0)

✦ Timeouts

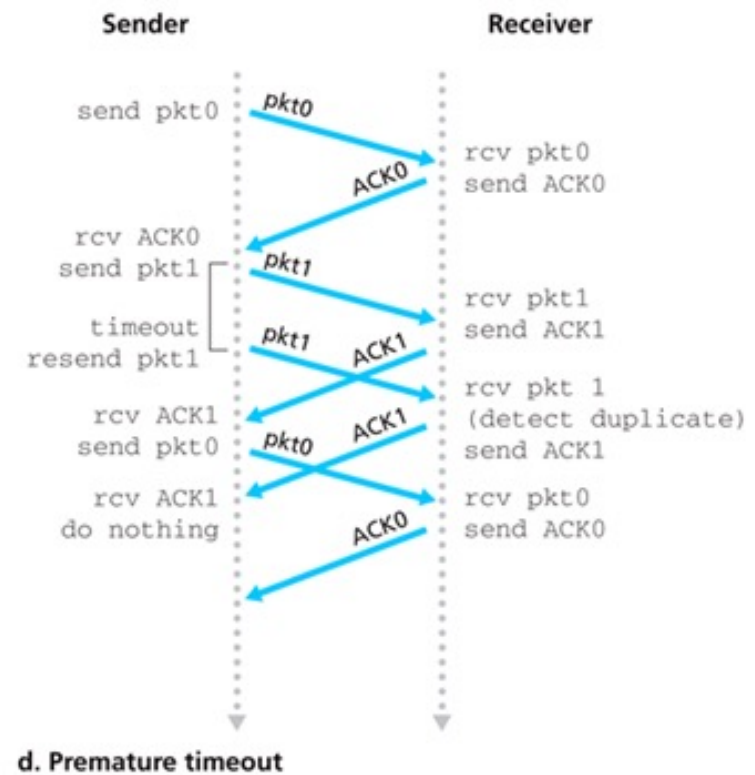
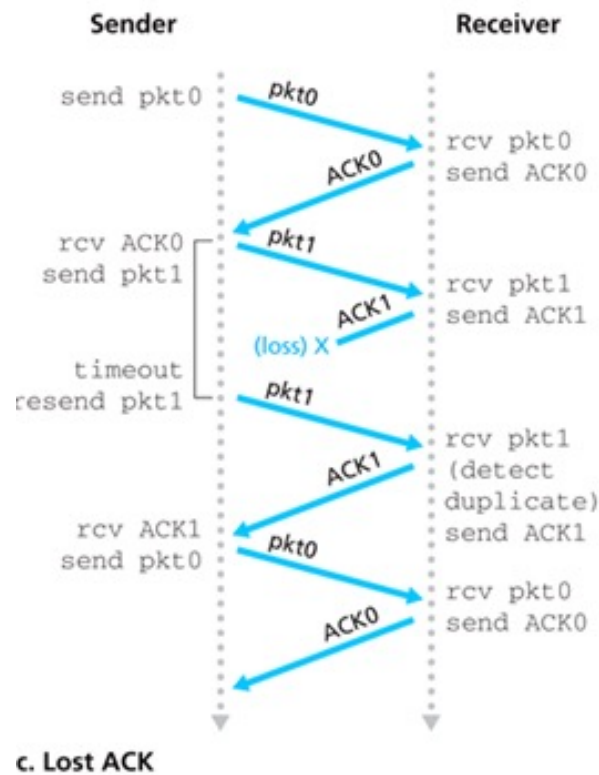
CONTROL DE FLUJO SIMPLEX

Sobre un canal ruidoso (pueden perderse datos)



CONTROL DE FLUJO SIMPLEX

Sobre un canal ruidoso (pueden perderse datos)



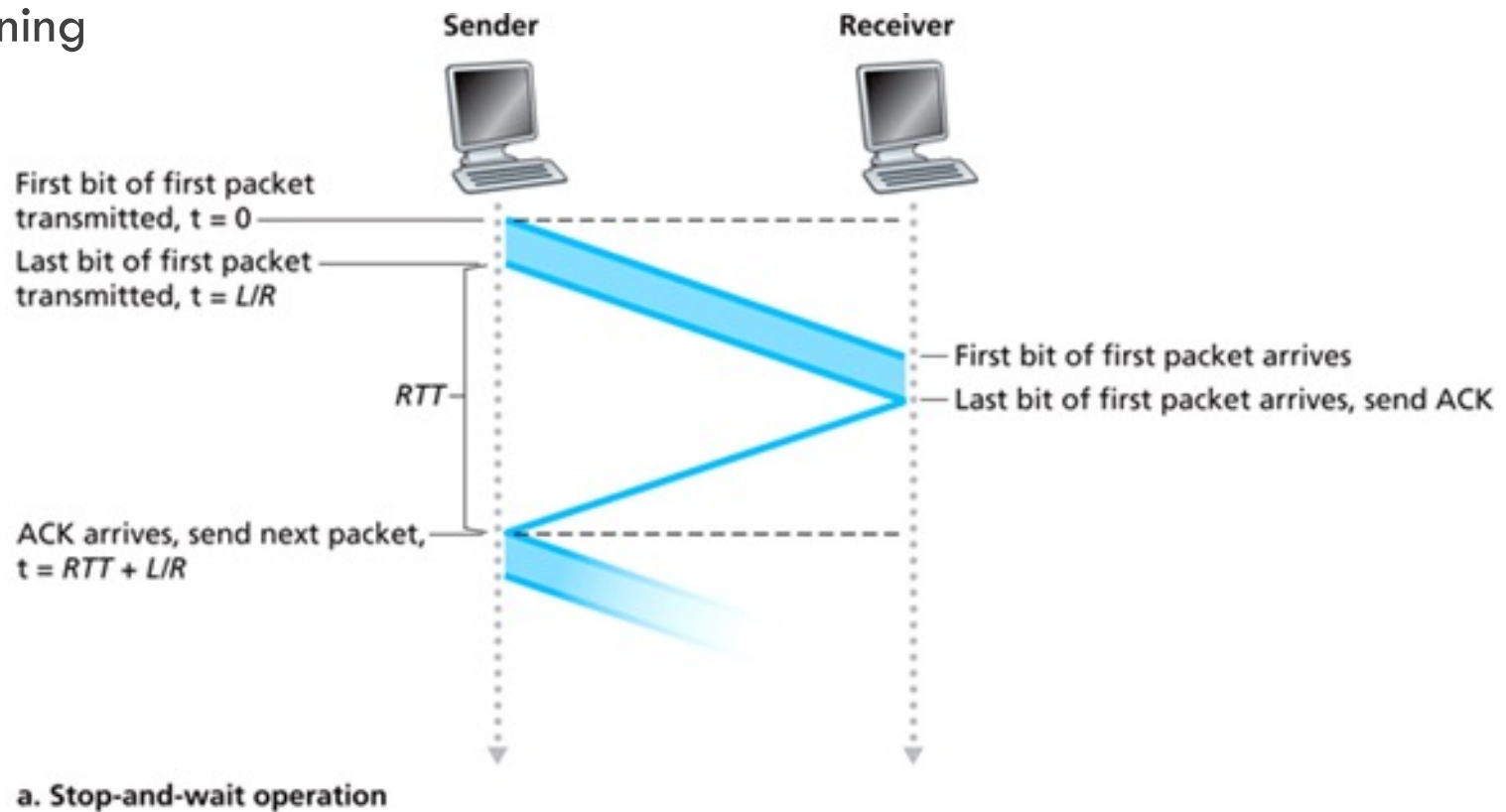
CONTROL DE FLUJO SLIDING WINDOW

- HALFDUPLEX/FULLDUPLEX
- PIPELINING
- PIGGY BACKING
- USO MÁS EFICIENTE DEL CANAL



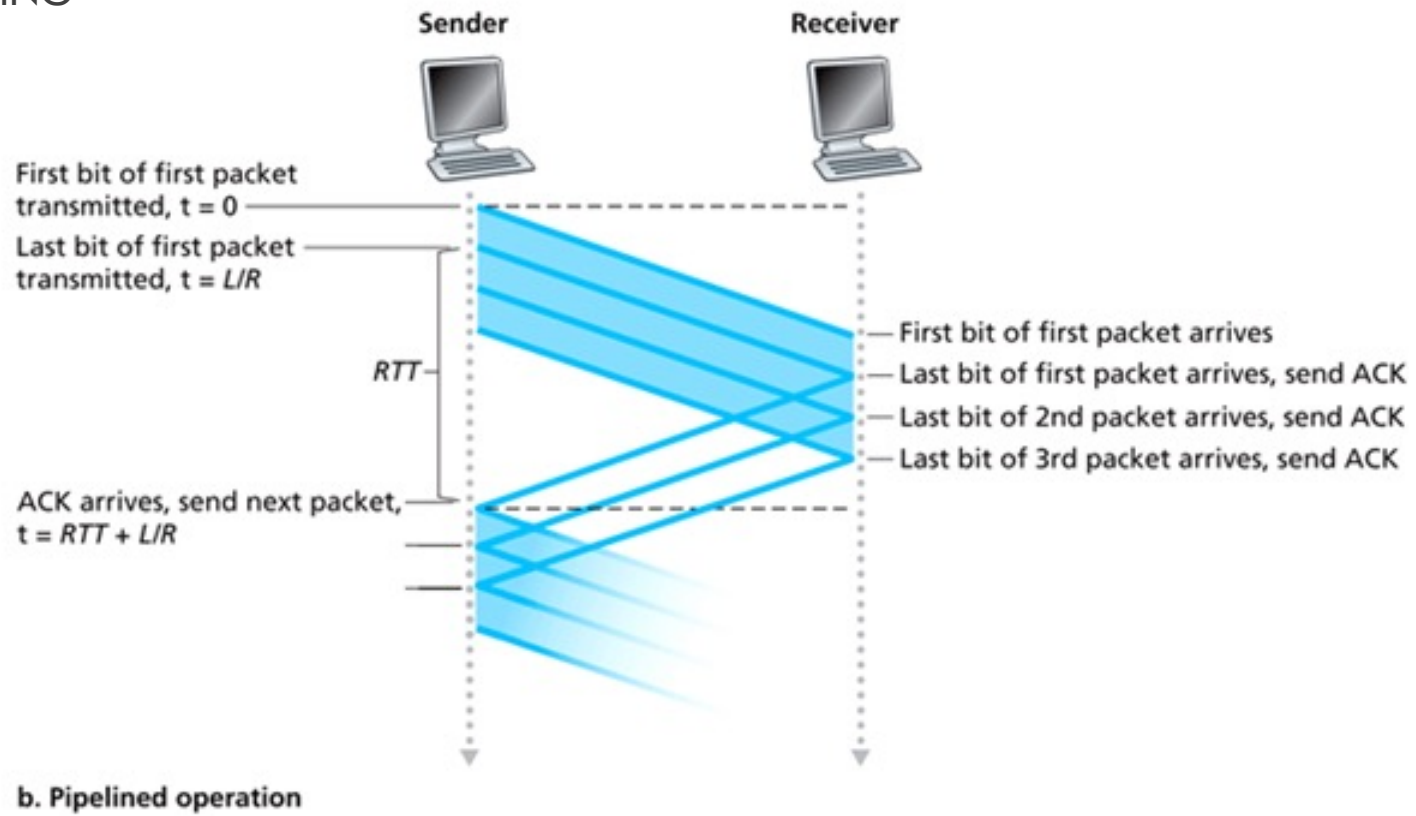
CONTROL DE FLUJO SLIDING WINDOWS

Pipelining



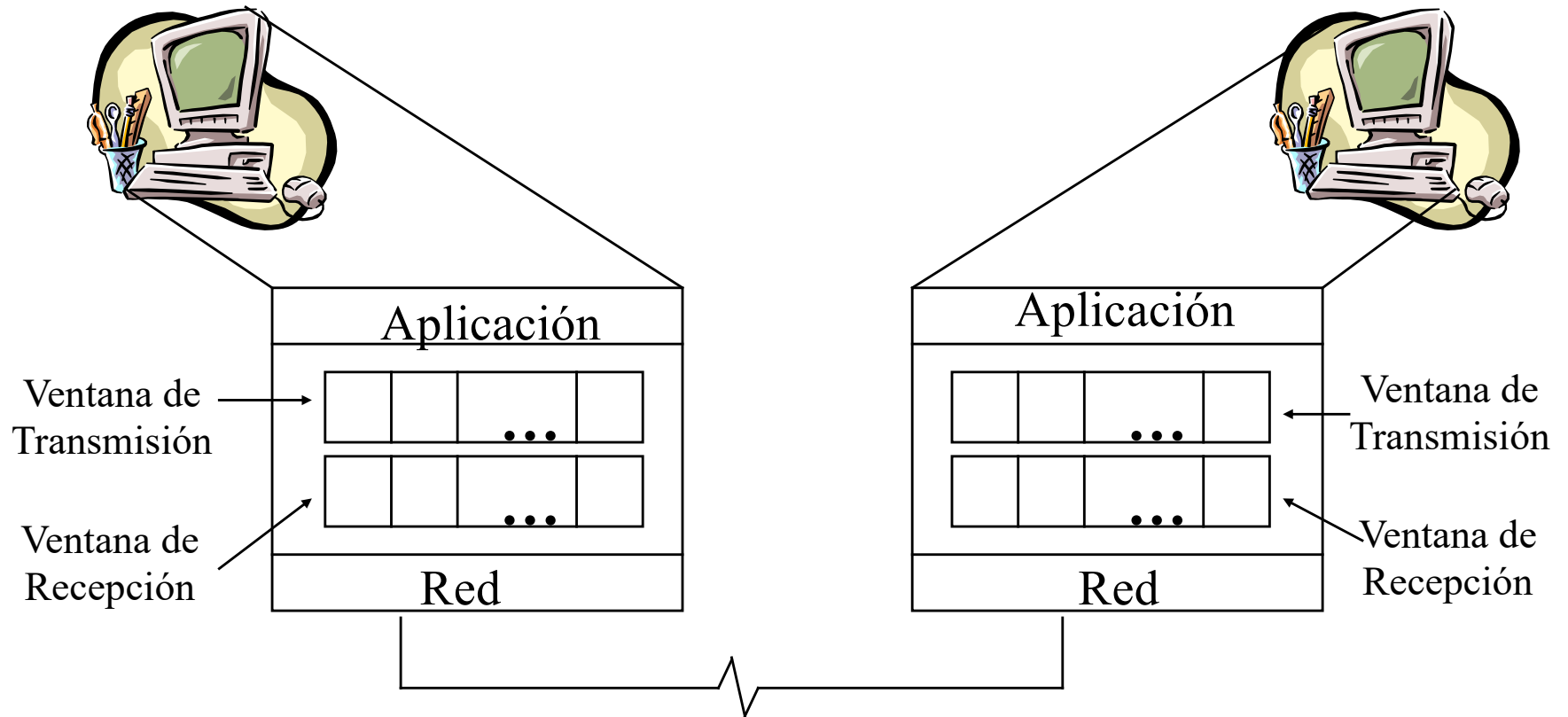
CONTROL DE FLUJO SLIDING WINDOW

● PIPELINING

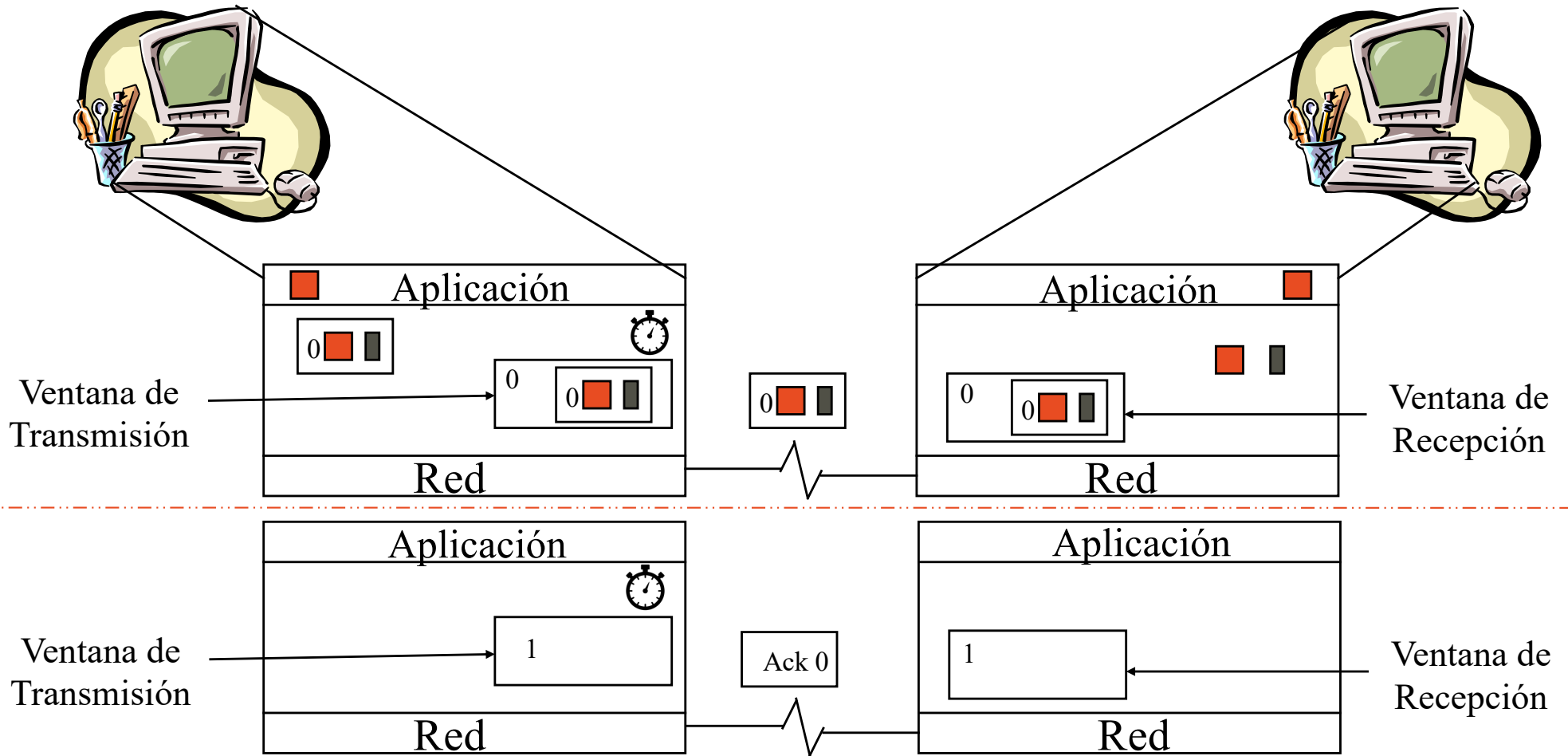


CONTROL DE FLUJO SLIDING WINDOW

- PIPELINING



CONTROL DE FLUJO SLIDING WINDOW



CONTROL DE FLUJO SLIDING WINDOWS



Go back n

- Ventana de recepción de tamaño 1.
- Si se daña un mensaje, descarta los demás de ese punto en adelante
- Números de secuencia =
 $0 - (\text{tamaño de la ventana de transmisión} - 1)$
- Se puede enviar Ack por cada mensaje, pero si se pierde uno de ellos no hay problema porque si llega otro con número de secuencia mayor se supone bien todo de ahí para atrás.

CONTROL DE FLUJO SLIDING WINDOWS

- GO BACK N

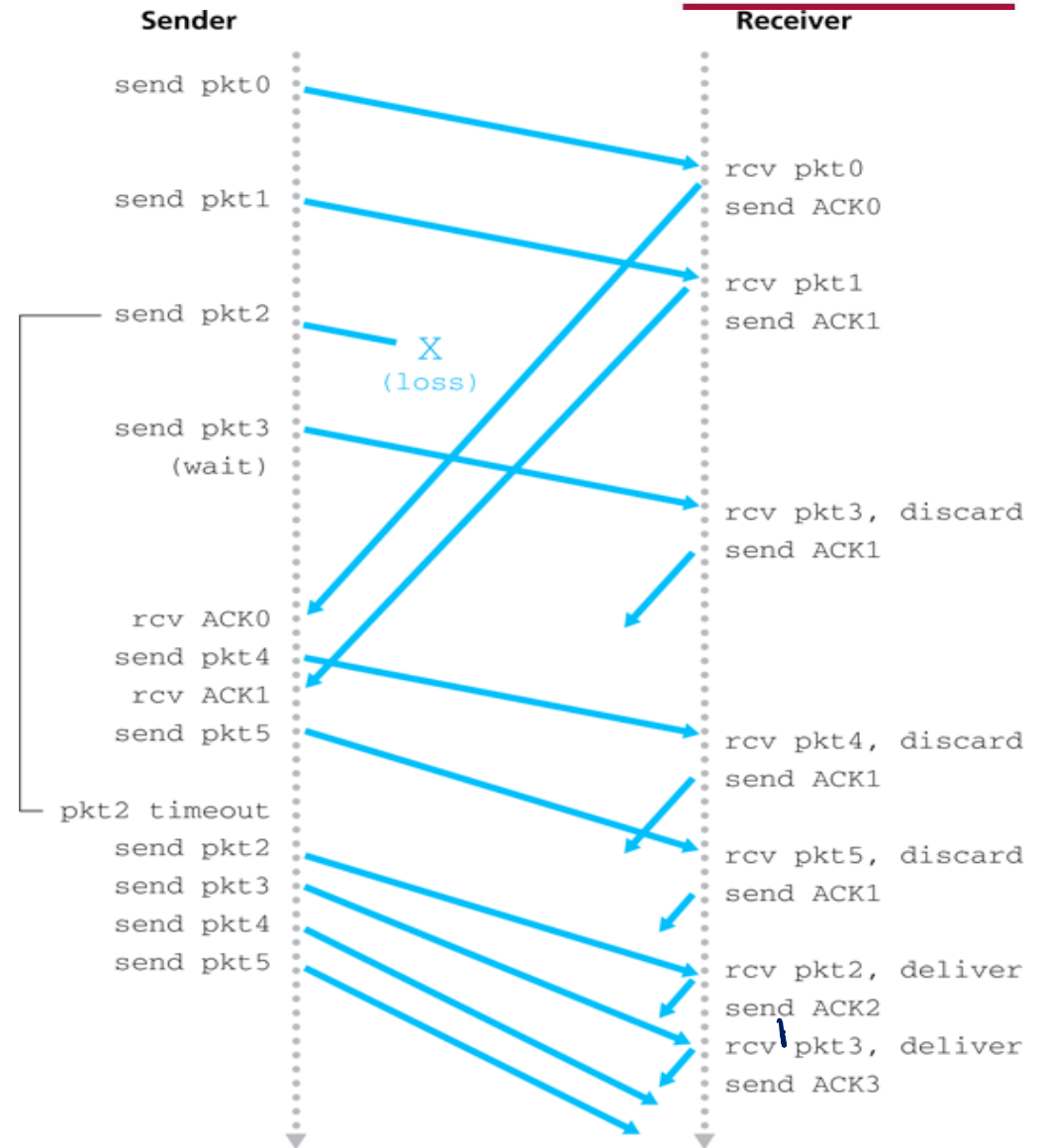


Figure 3.22 ♦ Go-Back-N in operation

CONTROL DE FLUJO SLIDING WINDOW

Go back n

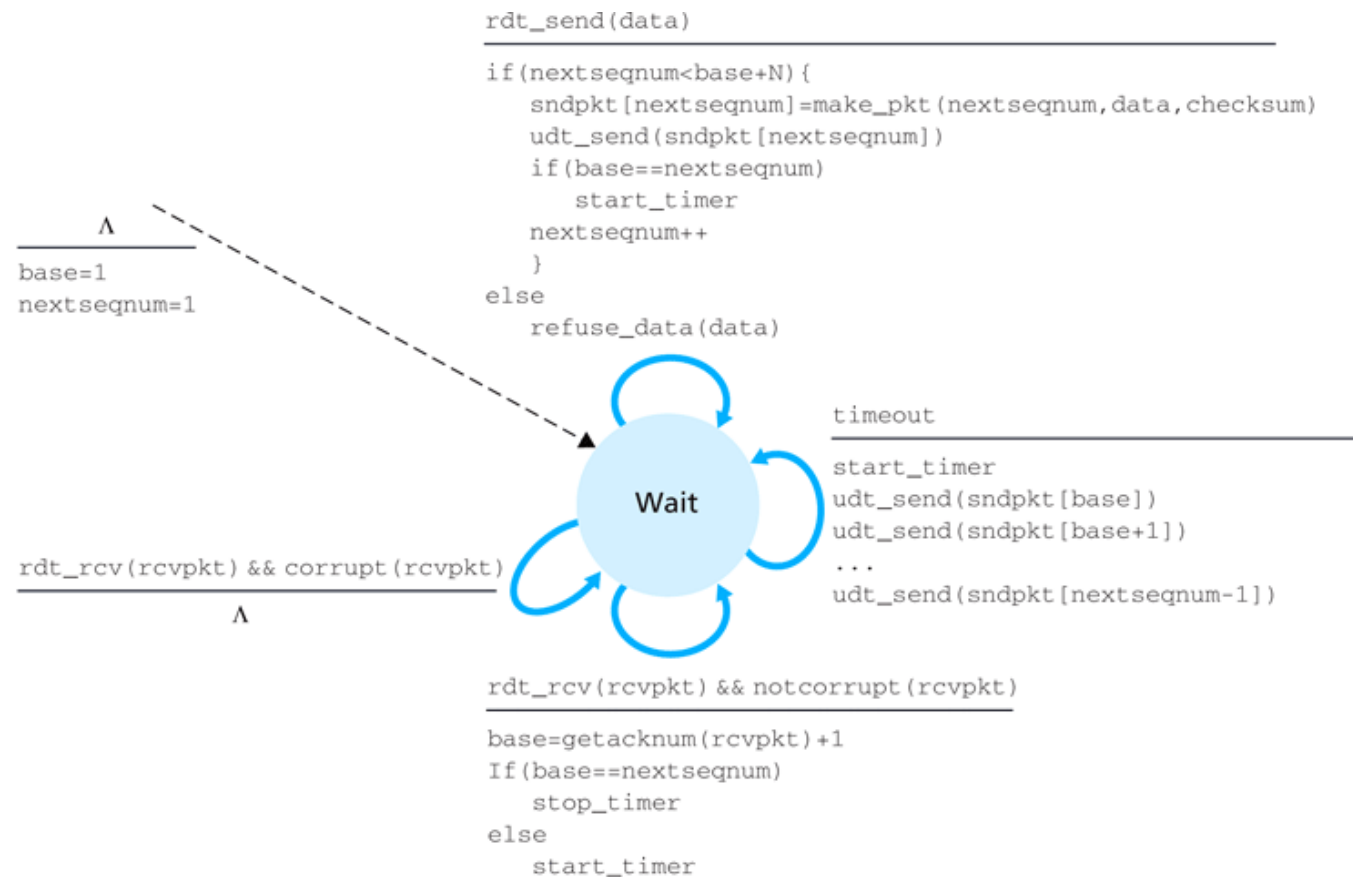


Figure 3.20 ♦ Extended FSM description of GBN sender

```

stateDiagram-v2
    [*] --> Wait : Lambda
    Wait --> Wait : default
    Wait --> Wait : udt_send(sndpkt)
  
```

Λ
 expectedseqnum=1
 sndpkt=make_pkt(0,ACK,checksum)

Figure 3.21 ♦ Extended FSM description of GBN receiver

CONTROL DE FLUJO SLIDING WINDOW



Repetición selectiva – selective repeat

- Ventana de recepción de tamaño n .
- Si se daña un mensaje, descarta solo el frame dañado y mantiene en la ventana los paquetes buenos recibidos después del dañado
- Espera recibir retransmisión del dañado y confirma todos los recibidos
- Números de secuencia entre 0 y $(\text{tamaño de la ventana} * 2) - 1$

CONTROL DE FLUJO SLIDING WINDOW

■ REPETICIÓN SELECTIVA

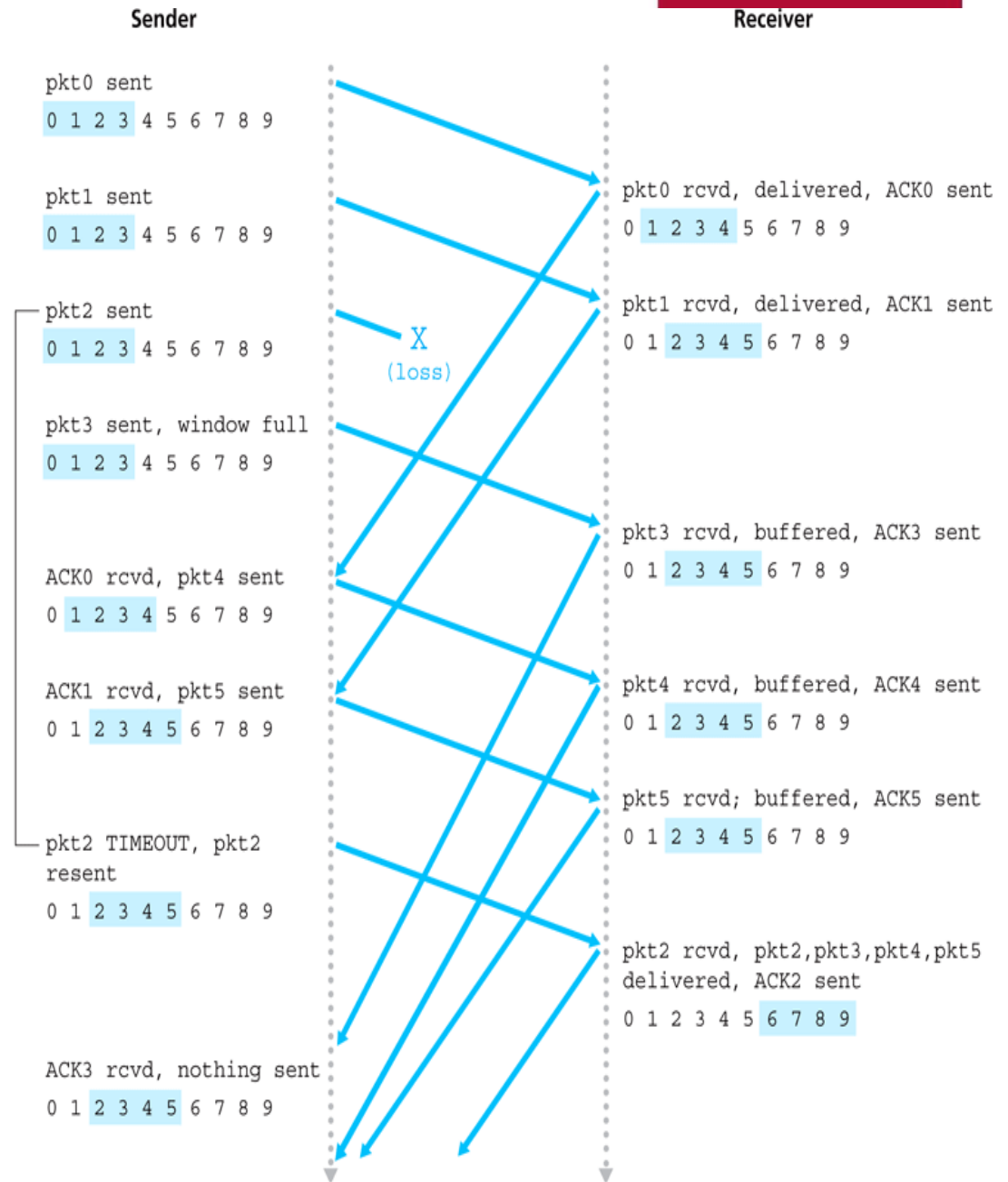


Figure 3.26 ♦ SR operation

CONTROL DE FLUJO SLIDING WINDOW

- REPETICIÓN SELECTIVA — SELECTIVE REPEAT

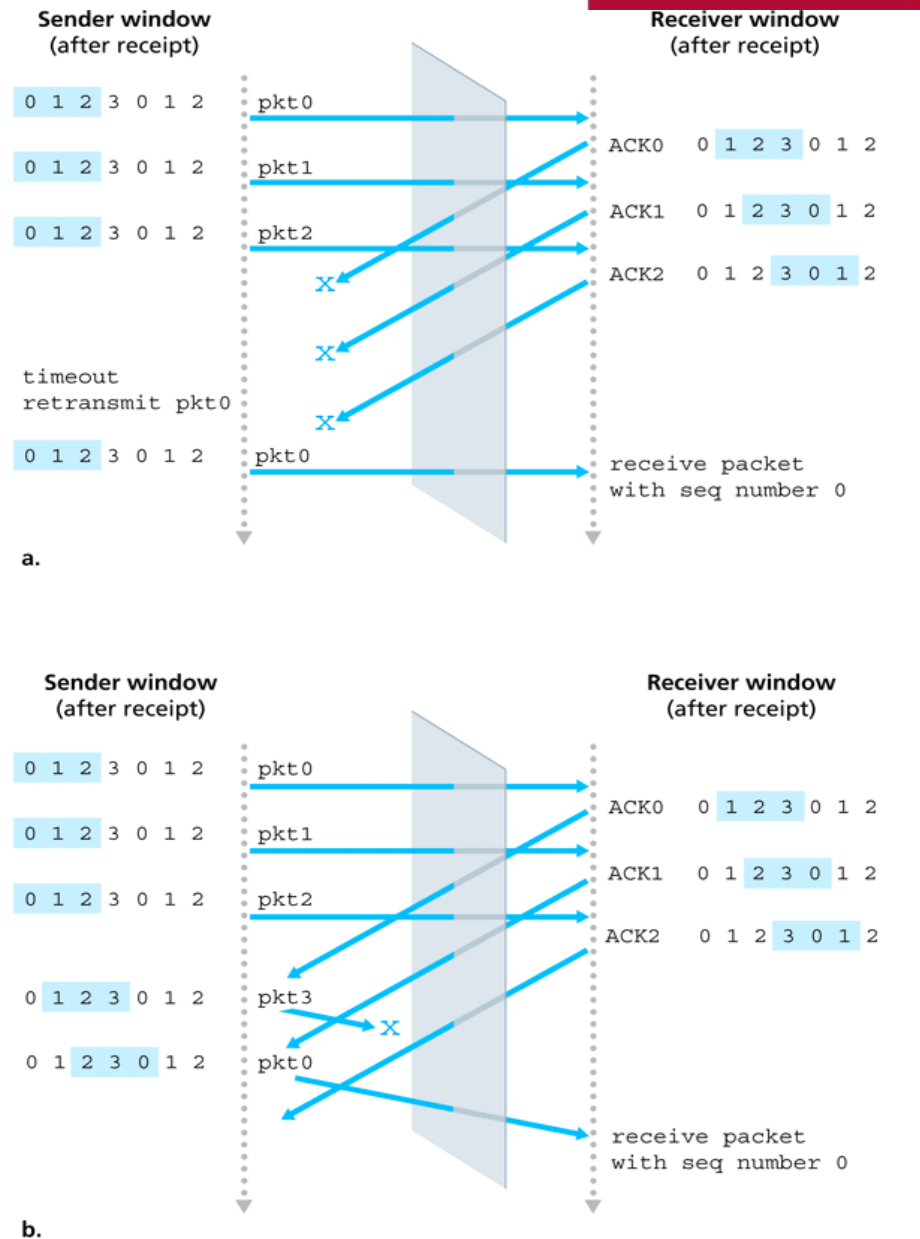


Figure 3.27 ♦ SR receiver dilemma with too-large windows: A new packet or a retransmission?

Computer Networking: A Top-Down Approach, 4/E. James F. Kurose, Keith W. Ross,

CONTROL DE FLUJO SLIDING WINDOW

• REPETICIÓN SELECTIVA — SELECTIVE REPEAT

Mechanism	Use, Comments
Checksum	Used to detect bit errors in a transmitted packet.
Timer	Used to timeout/retransmit a packet, possibly because the packet (or its ACK) was lost within the channel. Because timeouts can occur when a packet is delayed but not lost (premature timeout), or when a packet has been received by the receiver but the receiver-to-sender ACK has been lost, duplicate copies of a packet may be received by a receiver.
Sequence number	Used for sequential numbering of packets of data flowing from sender to receiver. Gaps in the sequence numbers of received packets allow the receiver to detect a lost packet. Packets with duplicate sequence numbers allow the receiver to detect duplicate copies of a packet.
Acknowledgment	Used by the receiver to tell the sender that a packet or set of packets has been received correctly. Acknowledgments will typically carry the sequence number of the packet or packets being acknowledged. Acknowledgments may be individual or cumulative, depending on the protocol.
Negative acknowledgment	Used by the receiver to tell the sender that a packet has not been received correctly. Negative acknowledgments will typically carry the sequence number of the packet that was not received correctly.
Window, pipelining	The sender may be restricted to sending only packets with sequence numbers that fall within a given range. By allowing multiple packets to be transmitted but not yet acknowledged, sender utilization can be increased over a stop-and-wait mode of operation. We'll see shortly that the window size may be set on the basis of the receiver's ability to receive and buffer messages, or the level of congestion in the network, or both.

Table 3.1 ♦ Summary of reliable data transfer mechanisms and their use

