Transport Layer

Dr Kunwar Pal

Dept. of Computer Science and Engineering

National Institute of Technology

Jalandhar, Punjab, India

kunwarp@nitj.ac.in

TRANSPORT-LAYER SERVICES

The transport layer is located between the network layer and the application layer.

The transport layer is responsible for providing services to the application layer; it receives services from the network layer.

- ✔ Process-to-Process Communication
- ✔ Addressing: Port Numbers
- ✔ Encapsulation and Decapsulation
- ✔ Multiplexing and Demultiplexing
- ✔ Flow Control
- ✔ Error Control
- ✔ Congestion Control
- ✔ Connectionless and Connection-Oriented Services

Figure 1 *Network layer versus transport layer*

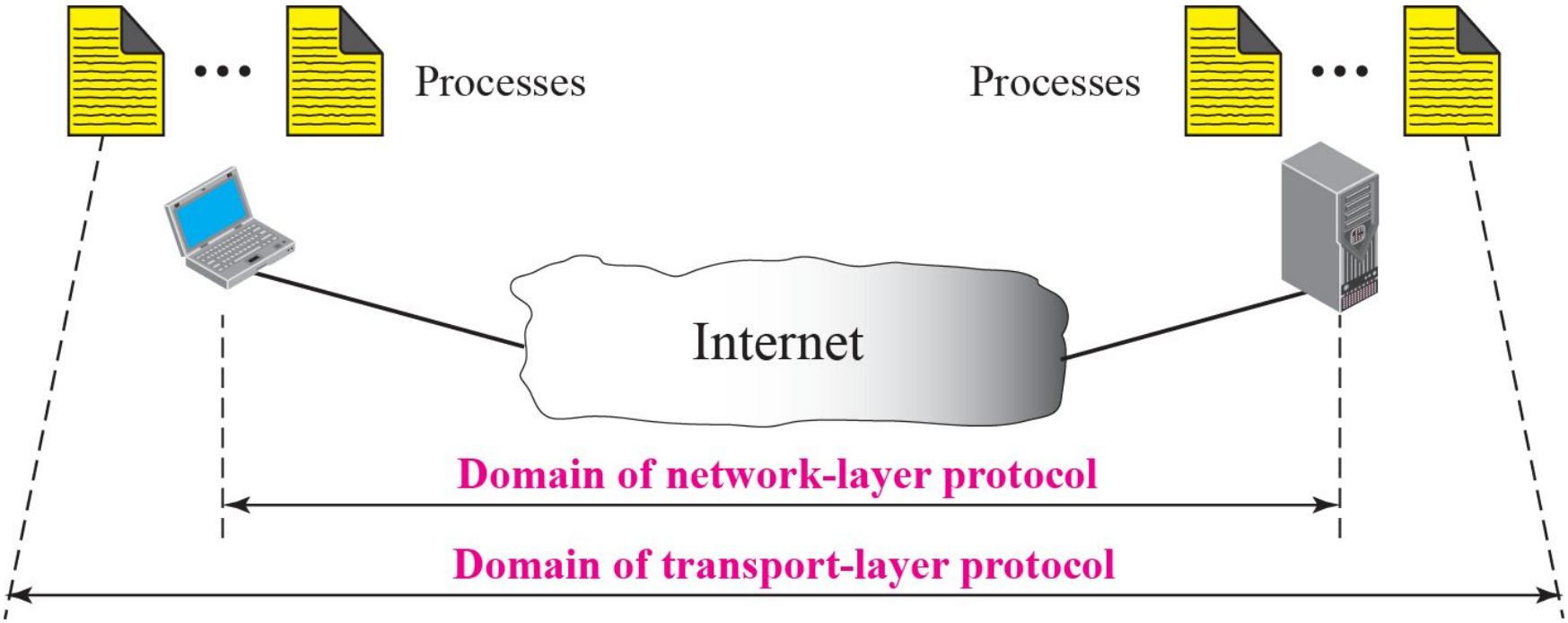


Figure 2 *Port numbers*

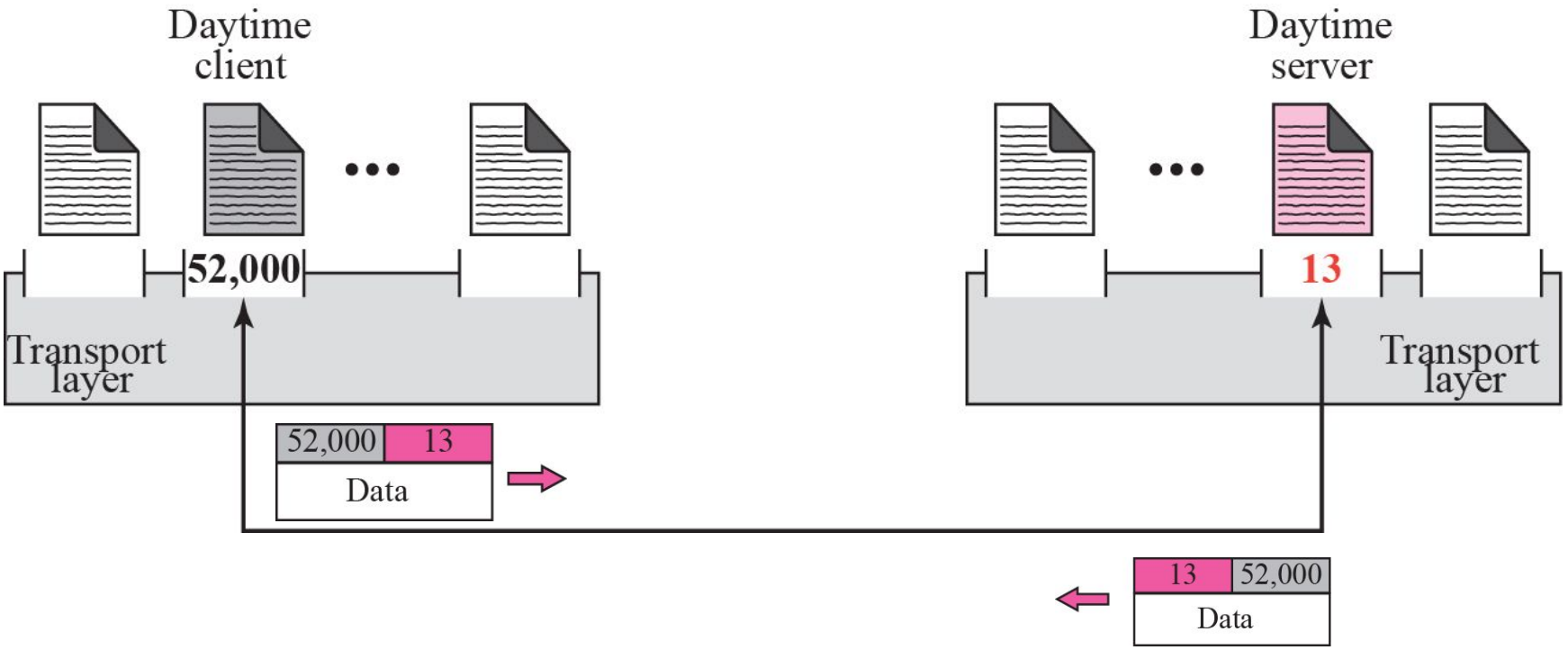


Figure 3 *IP addresses versus port numbers*

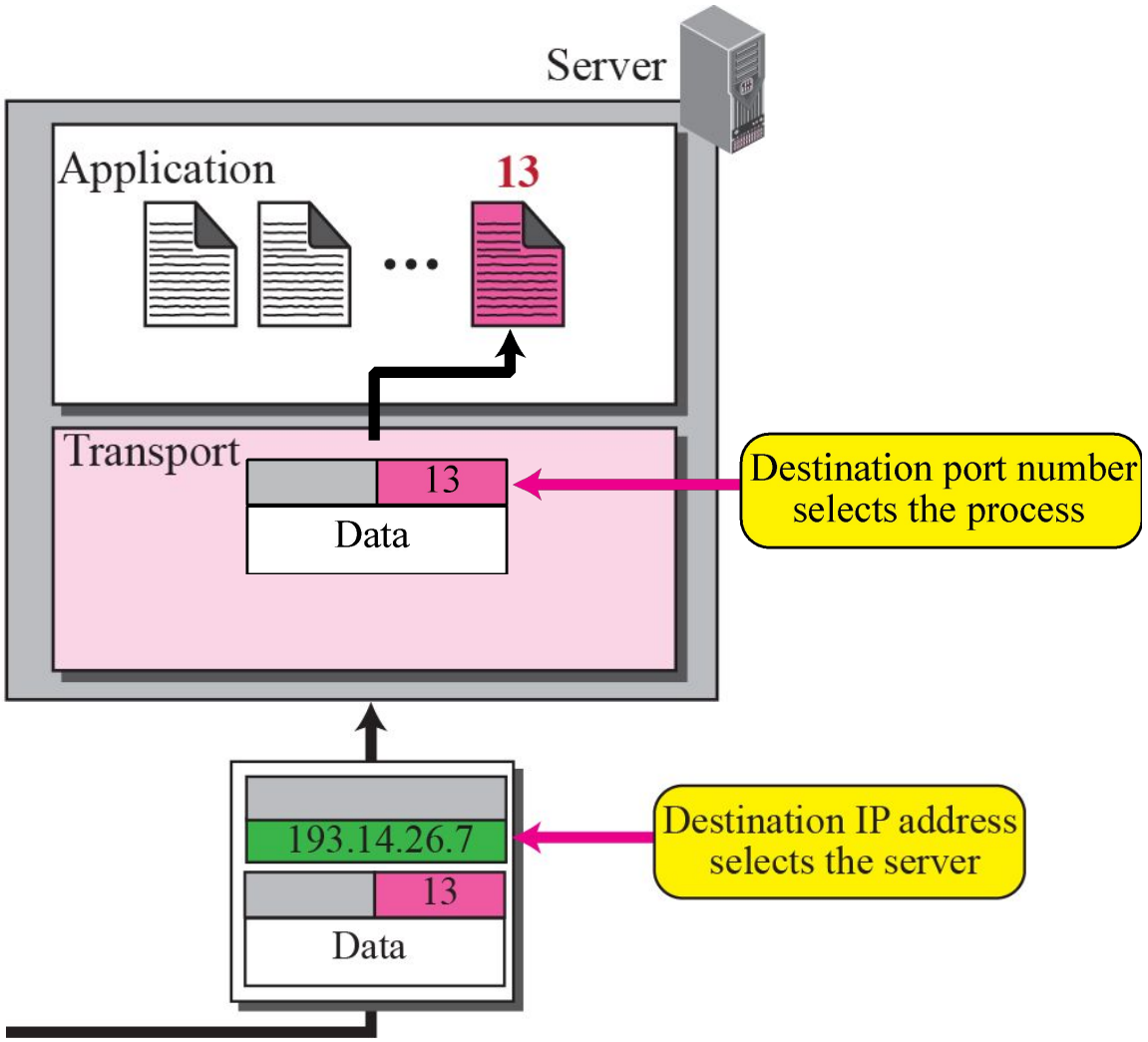
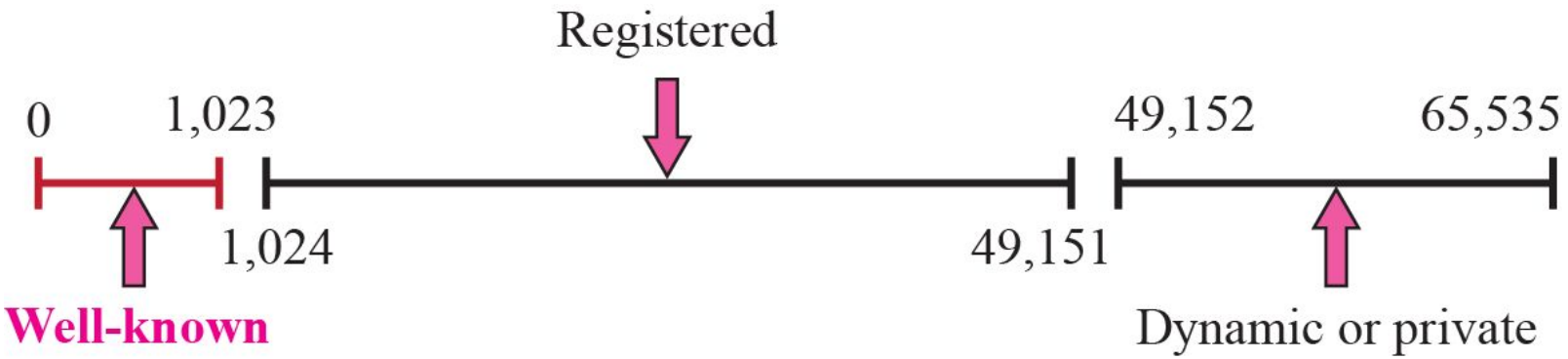


Figure 4 *ICANN ranges*

ICANN: Internet Corporation for Assigned Names and Numbers



Well-known ports. The ports ranging from 0 to 1,023 are assigned and controlled by ICANN. These are the well-known ports.

Registered ports. The ports ranging from 1,024 to 49,151 are not assigned or controlled by ICANN. They can only be registered with ICANN to prevent duplication.

Dynamic ports. The ports ranging from 49,152 to 65,535 are neither controlled nor registered. They can be used as temporary or private port numbers.

Figure 5 *Socket address*

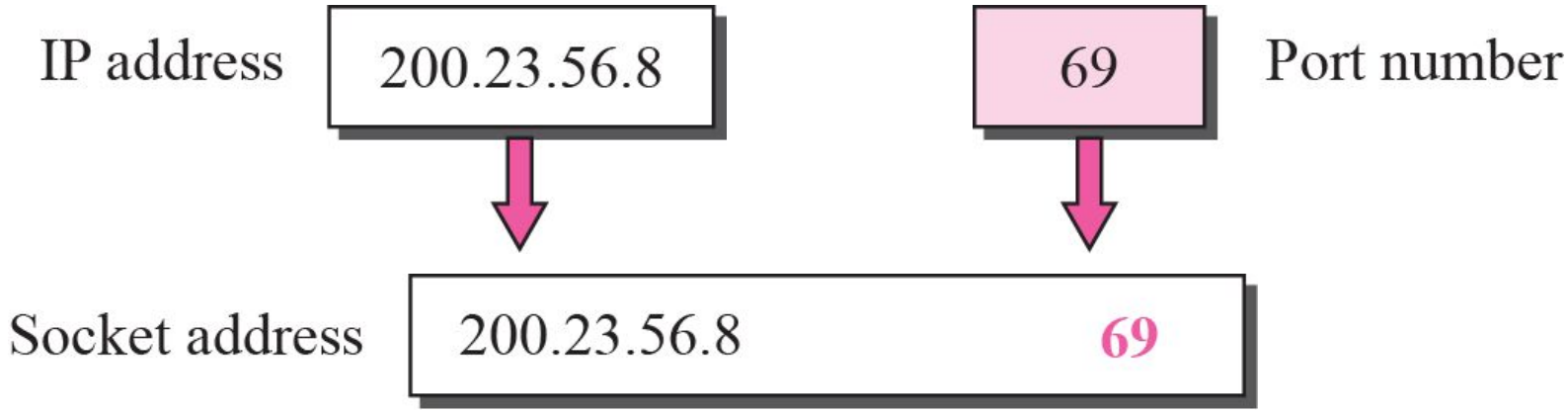


Figure 6 *Encapsulation and decapsulation*

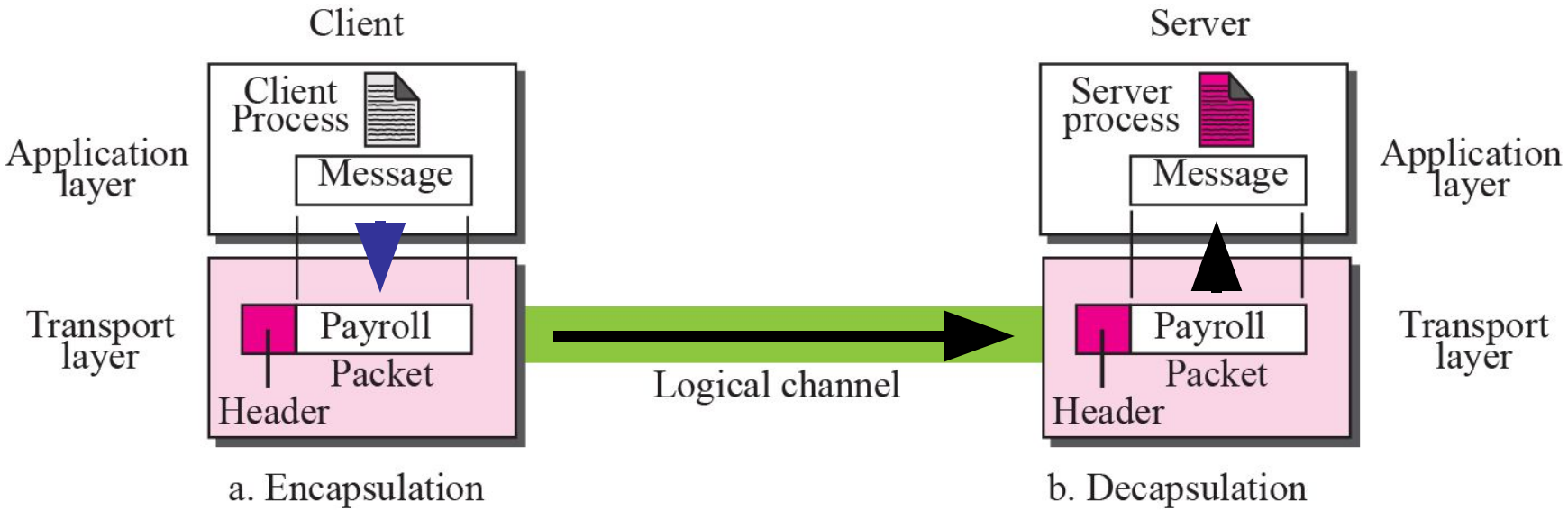


Figure 7 *Multiplexing and demultiplexing*

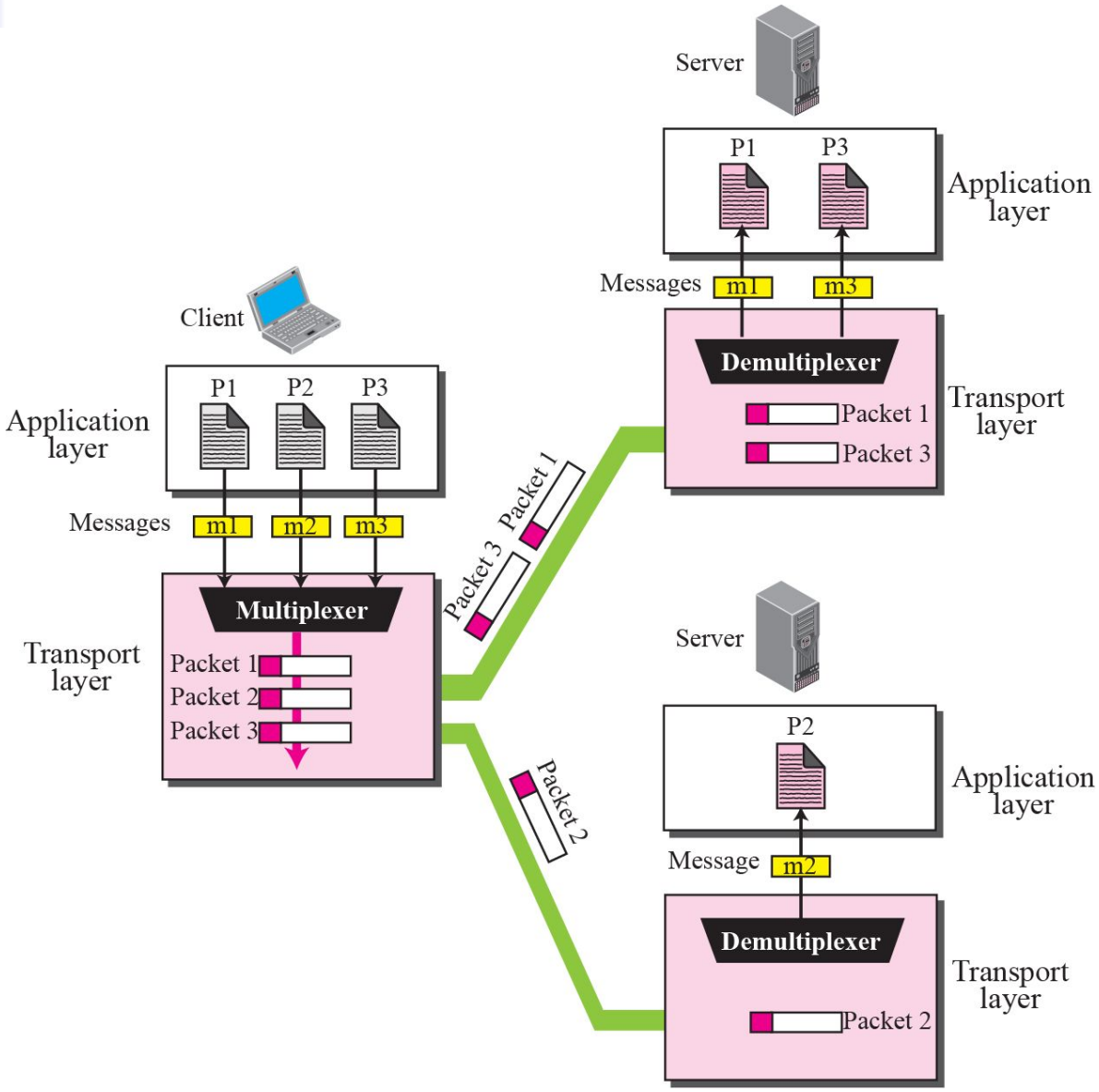


Figure 8 *Pushing or pulling*

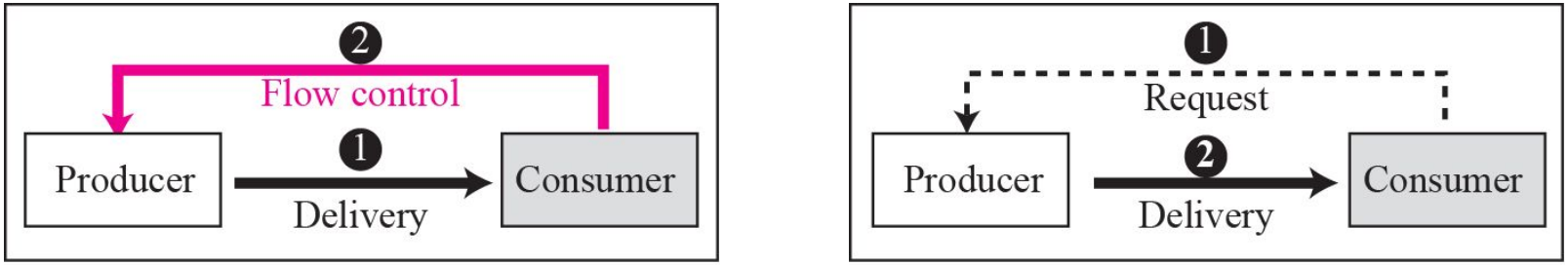


Figure 9 *Flow control at the transport layer*

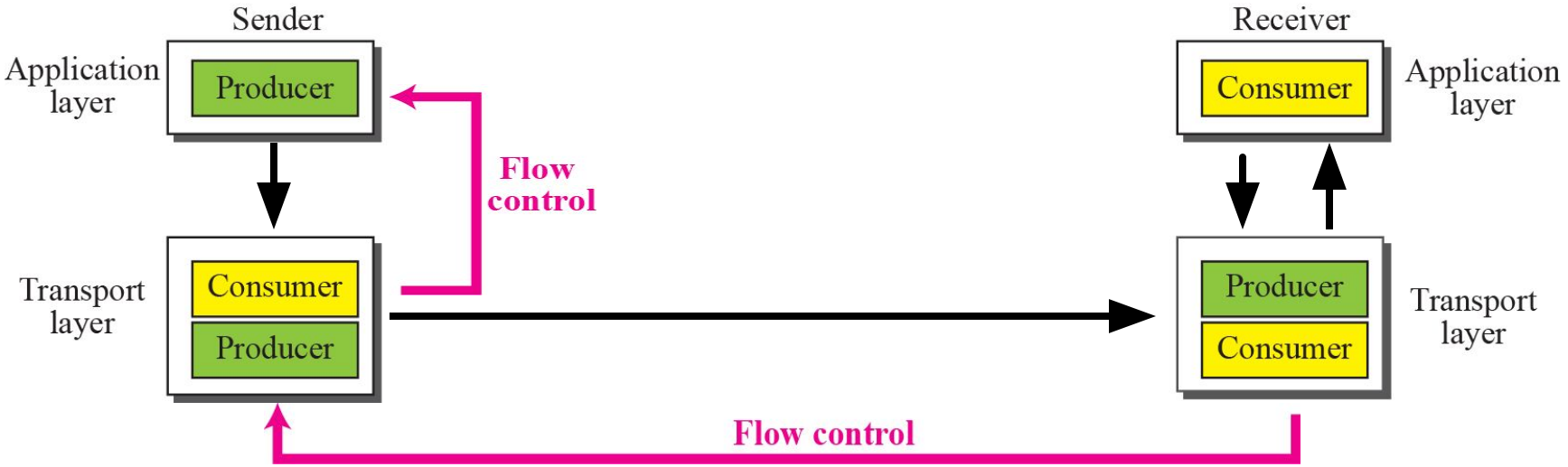
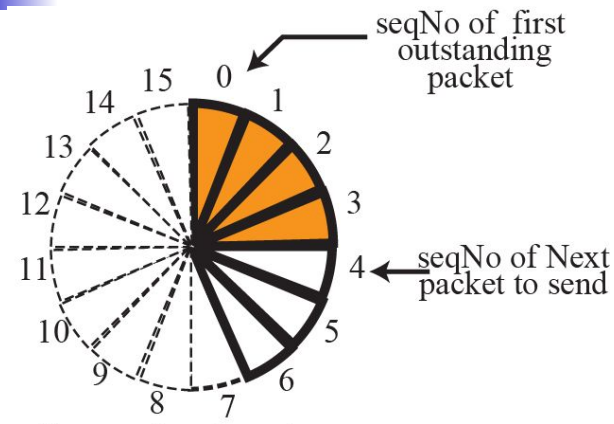


Figure 10 *Error control at the transport layer*

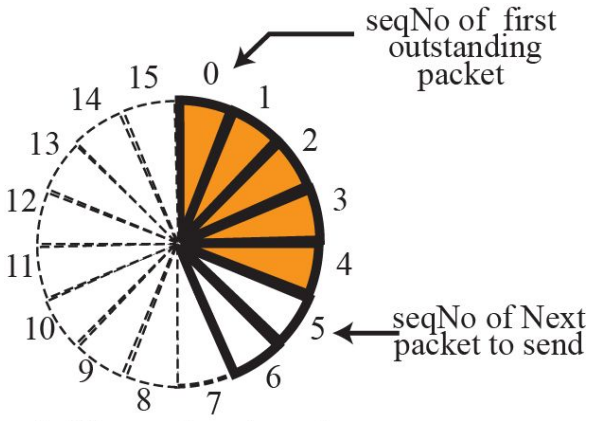
Note

For error control, the sequence numbers are modulo 2^m, where m is the size of the sequence number field in bits.

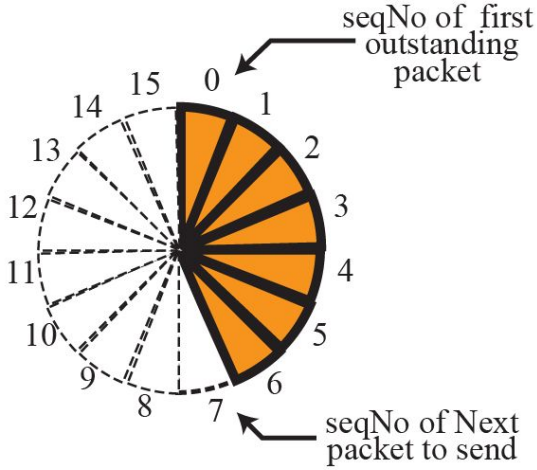
Figure 11 *Sliding window in circular format*



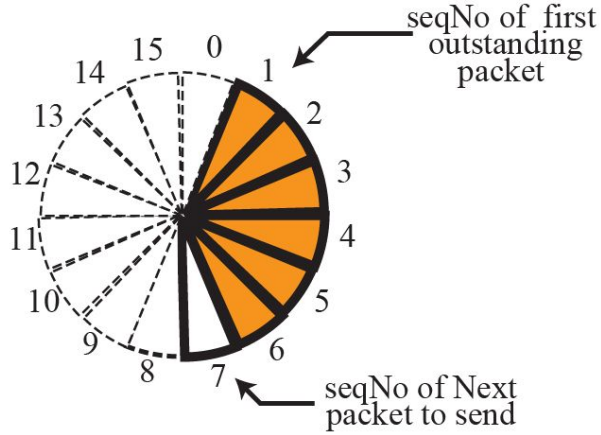
a. Four packets have been sent



b. Five packets have been sent



c. Seven packets have been sent
window is full



d. Packet 0 has been acknowledged,
window slides

Figure 12 *Sliding window in linear format*

a. Four packets have been sent

b. Five packets have been sent

c. Seven packets have been sent
window is full

d. Packet 0 have been acknowledged
and window slid

- ***Congestion in a network may occur if the load on the network-***
  The number of packet sent to the network is greater than the capacity of the network.

- ***Congestion control is a mechanisms and techniques to control the congestion and keep the load below the capacity.***

- ***Congestion control refers to techniques and mechanisms that can either prevent congestion, before it happens, or remove congestion, after it has happened.***

- ***In general, we can divide congestion control mechanisms into two broad categories: open-loop congestion control (prevention) and closed-loop congestion control (removal).***

Figure 13 *Connectionless service*

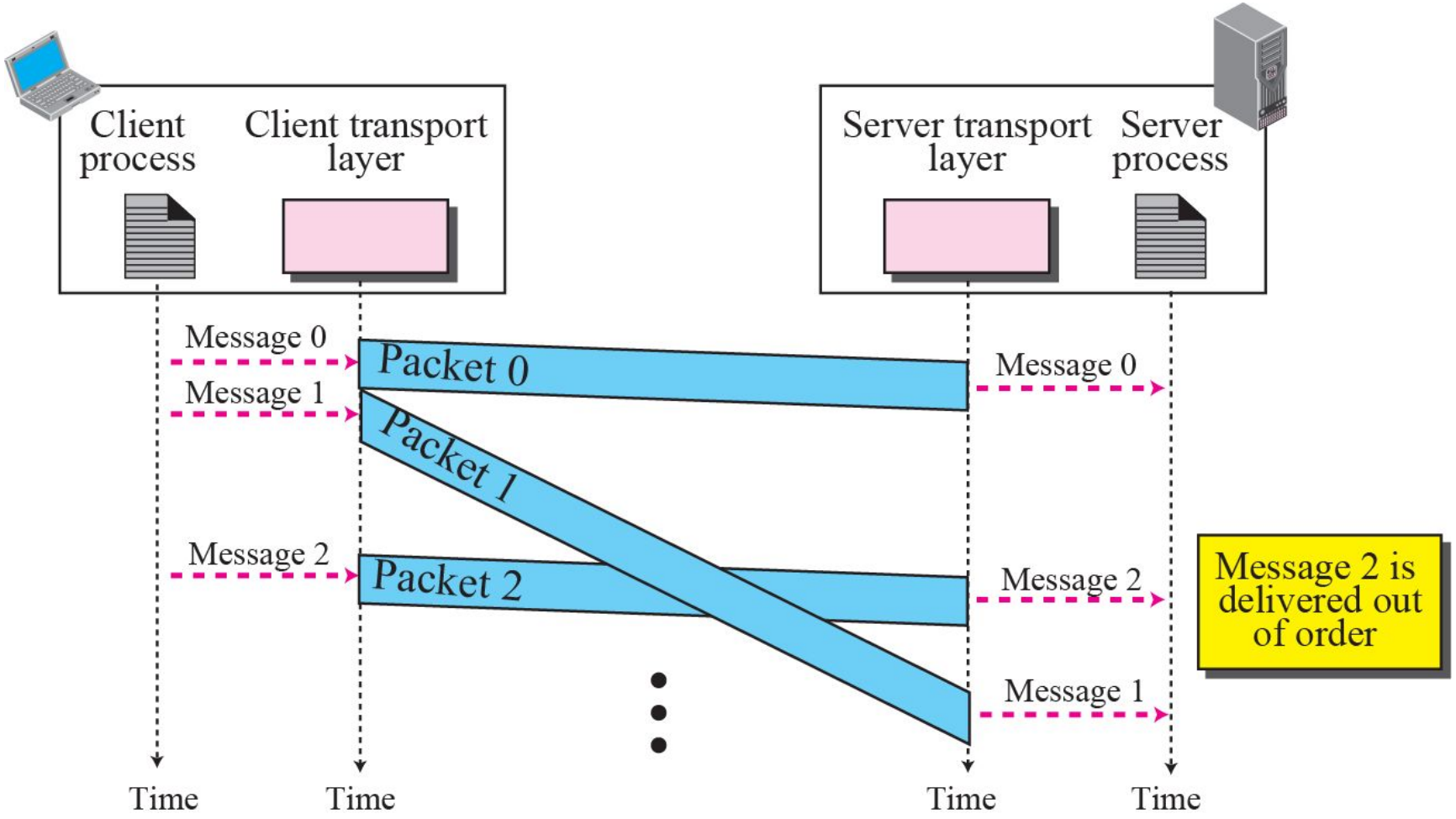


Figure 13 *Connection-oriented service*

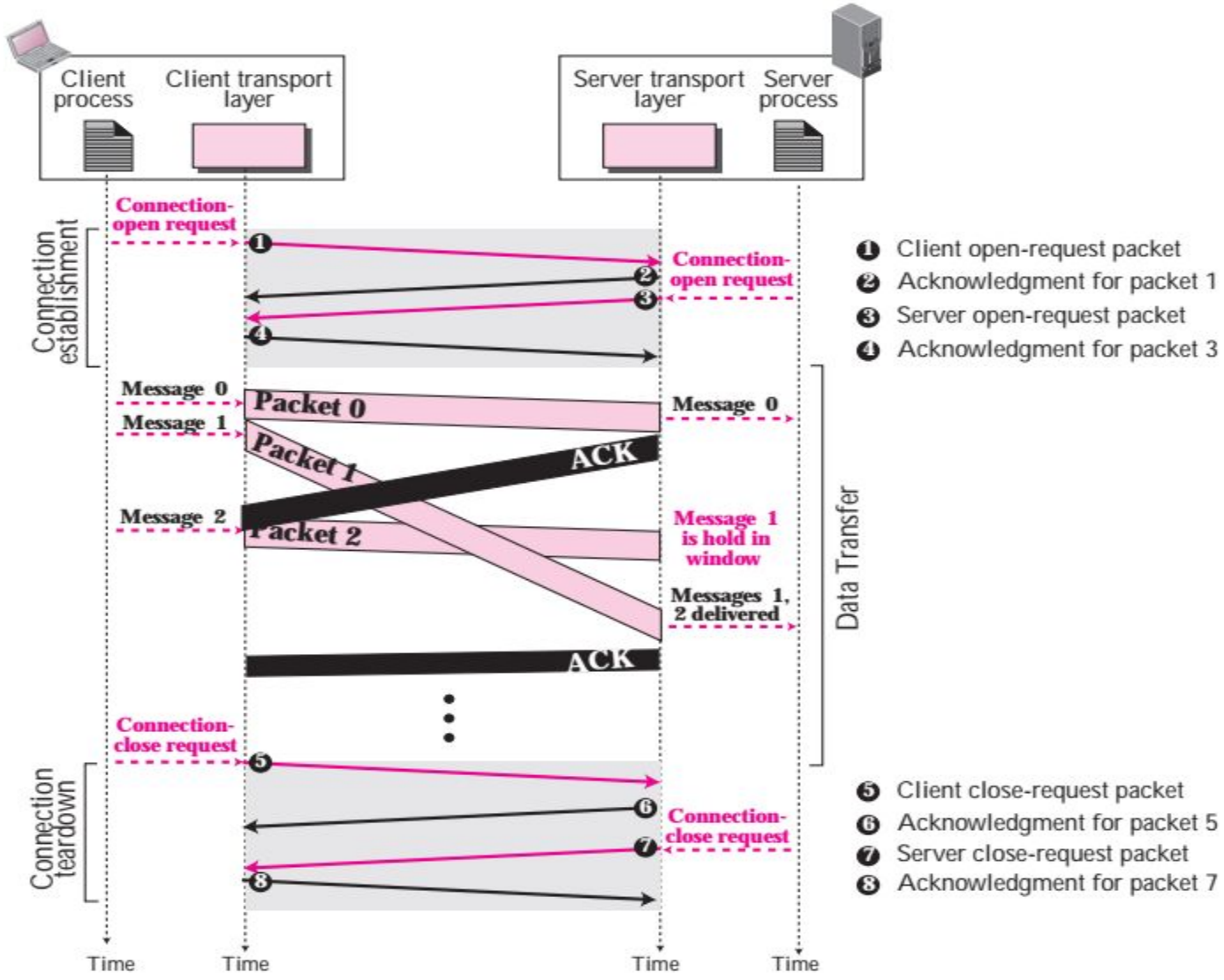
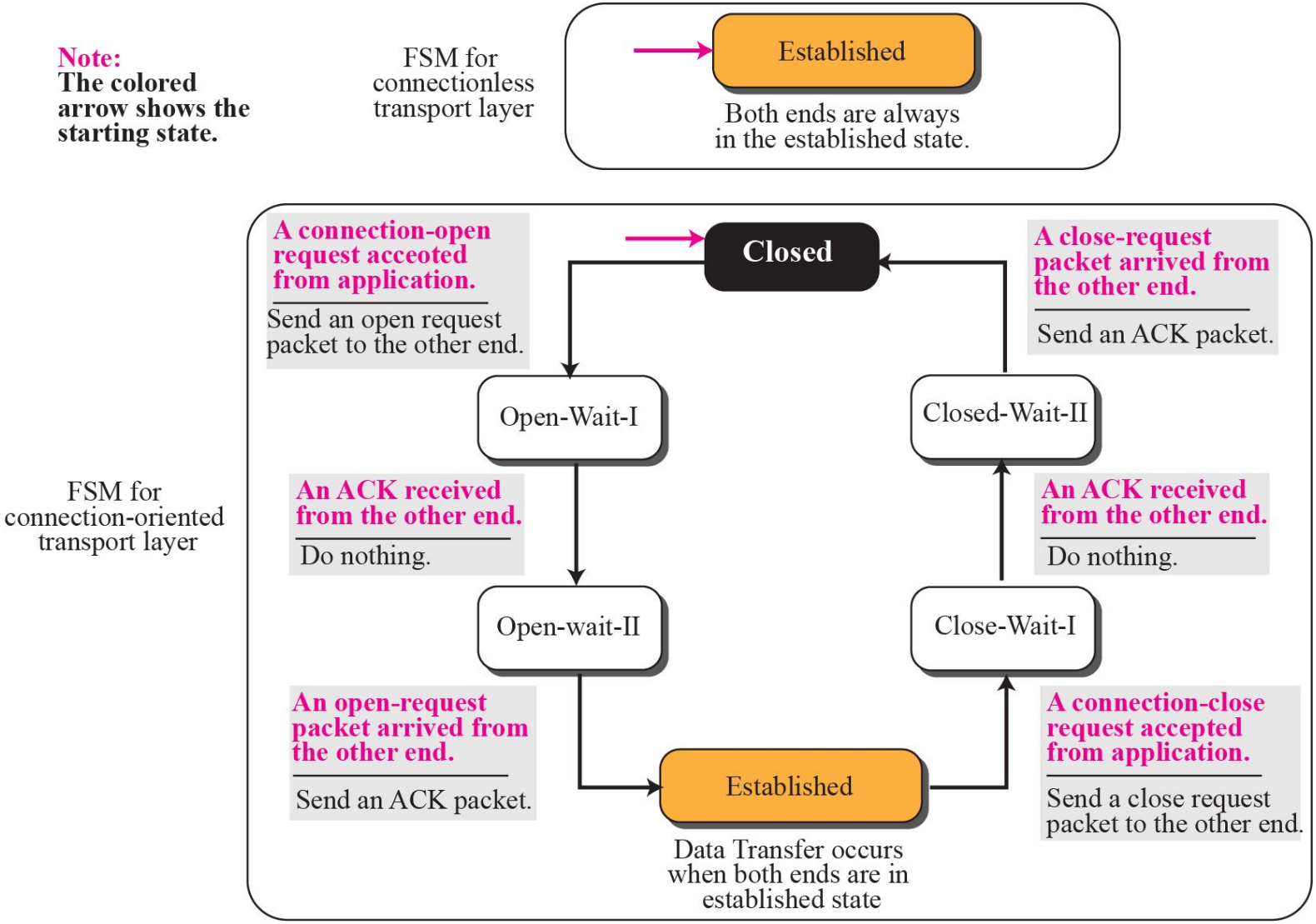


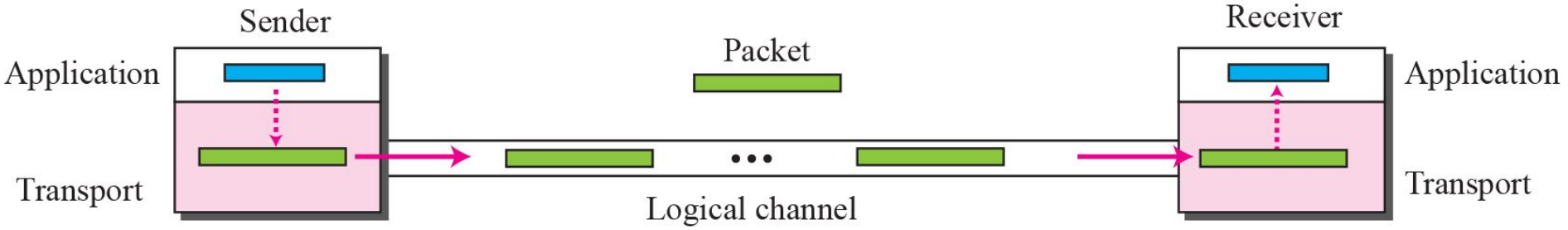
Figure 15 *Connectionless and connection-oriented services as FSMs*



TRANSPORT-LAYER PROTOCOLS

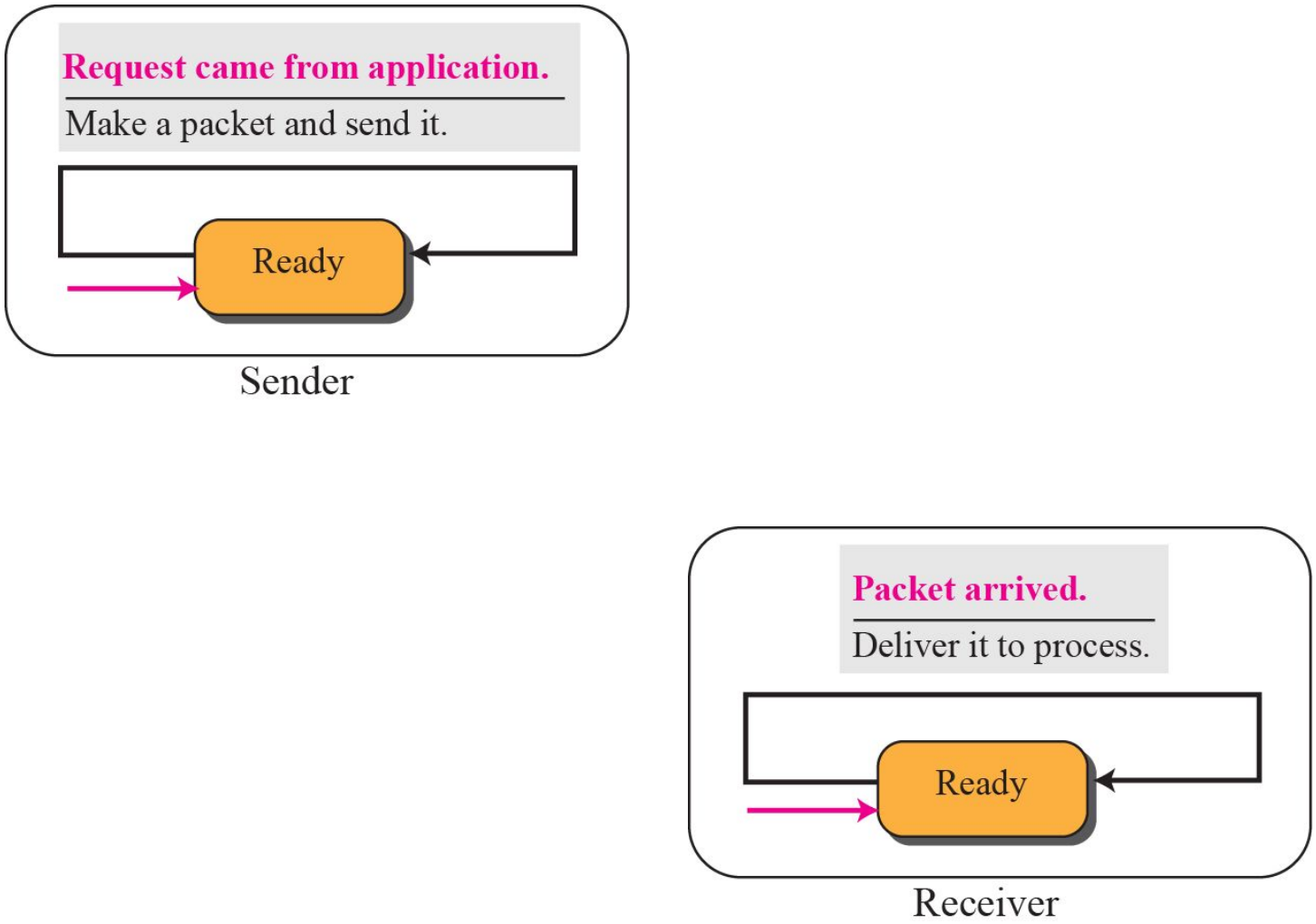
- ✔ **Simple Protocol**
- ✔ **Stop-and-Wait Protocol**
- ✔ **Go-Back-*N* Protocol**
- ✔ **Selective-Repeat Protocol**
- ✔ **Bidirectional Protocols: Piggybacking**

Figure 16 *Simple protocol*



- Each FSM has only one state, the *ready state*.
- The sending machine remains in the ready state until a request comes from the process in the application layer.
- When this event occurs, the sending machine encapsulates the message in a packet and sends it to the receiving machine.
- The receiving machine remains in the ready state until a packet arrives from the sending machine.
- When this event occurs, the receiving machine decapsulates the message out of the packet and delivers it to the process at the application layer.

Figure 17 *FSMs for simple protocol*



Note

The simple protocol is a connectionless protocol that provides neither flow nor error control.

Figure shows an example of communication using this protocol. It is very simple. The sender sends packets one after another without even thinking about the receiver.

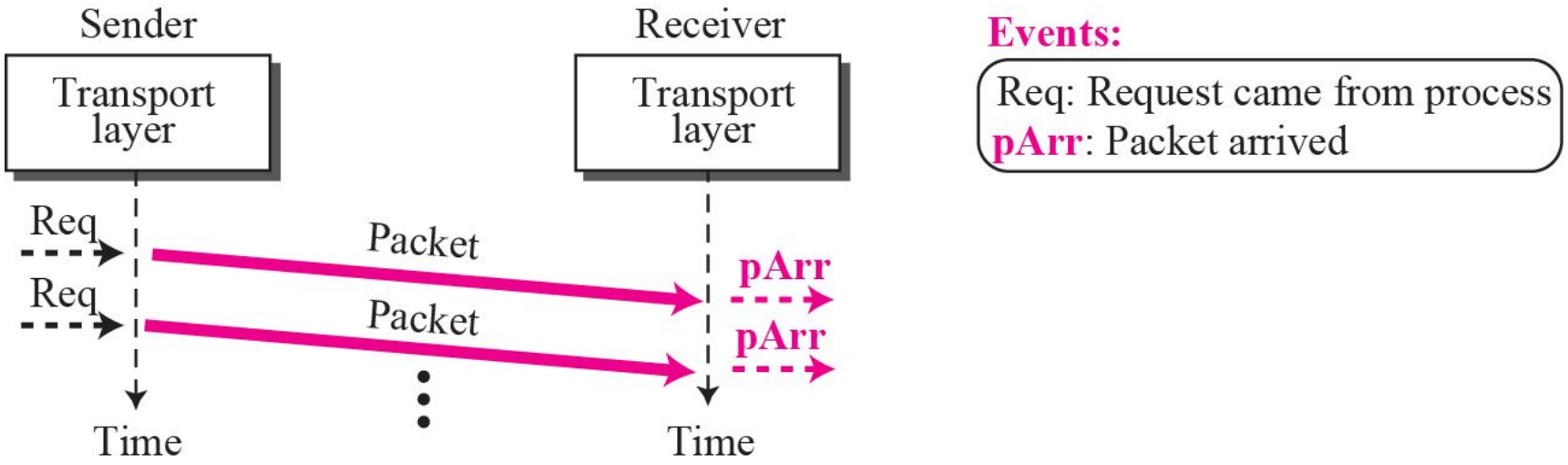
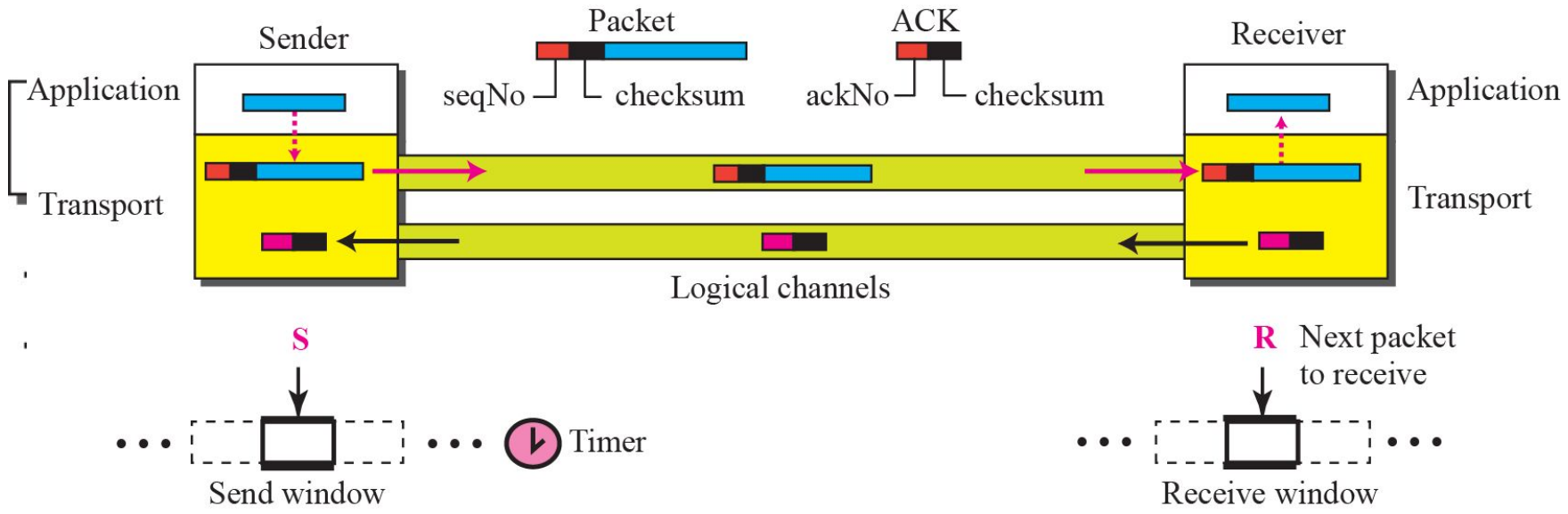


Figure 19 *Stop-and-wait protocol*



Note

In Stop-and-Wait protocol, flow control is achieved by forcing the sender to wait for an acknowledgment, and error control is achieved by discarding corrupted packets and letting the sender resend unacknowledged packets when the timer expires.

Note

In the Stop-and-Wait protocol, we can use a 1-bit field to number the packets. The sequence numbers are based on modulo-2 arithmetic.

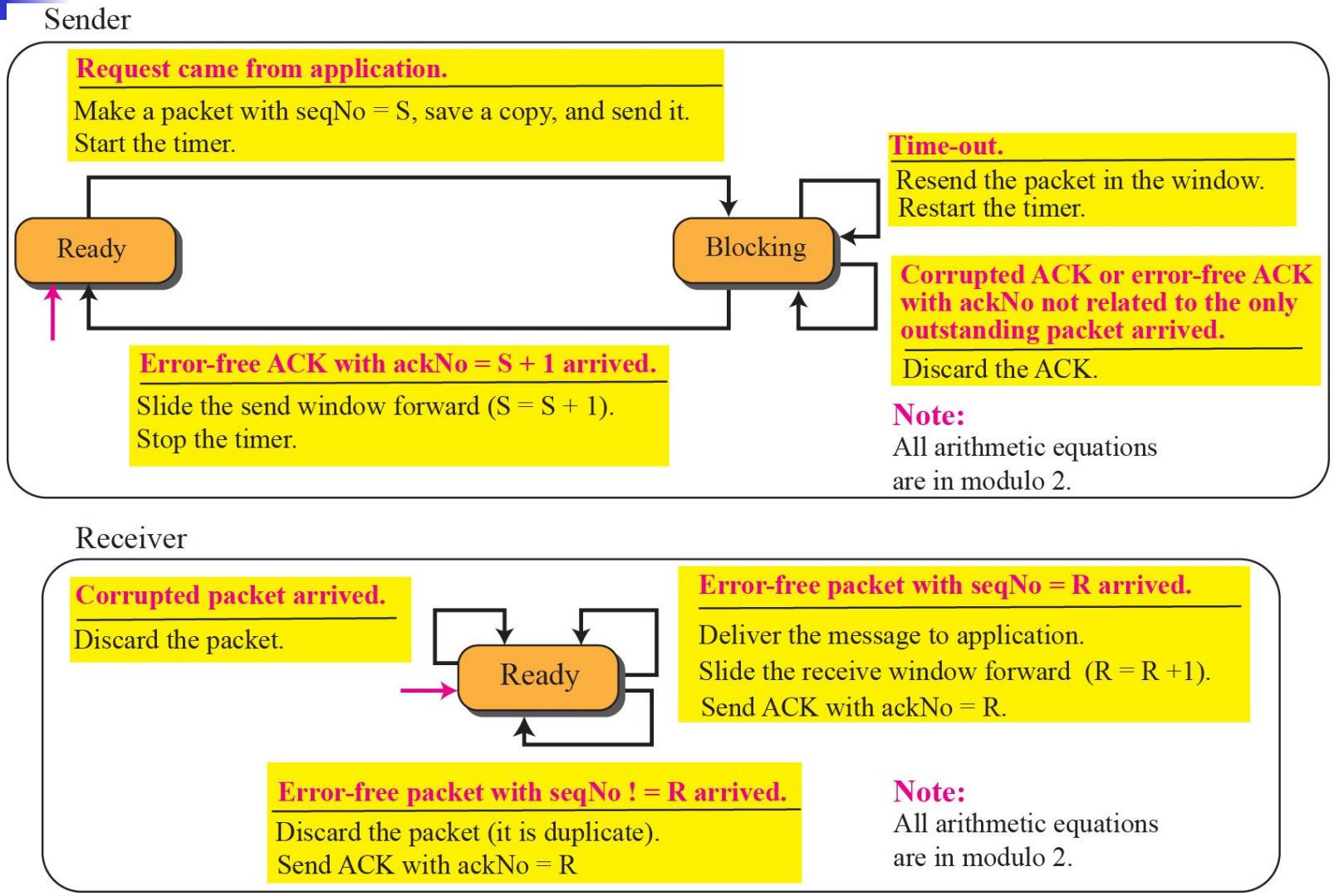
Note

In the Stop-and-Wait protocol, the acknowledgment number always announces in modulo-2 arithmetic the sequence number of the next packet expected.

Note

All calculation in the Stop-and-Wait protocol is in modulo 2.

Figure 20 *FSMs for stop-and-wait protocol*

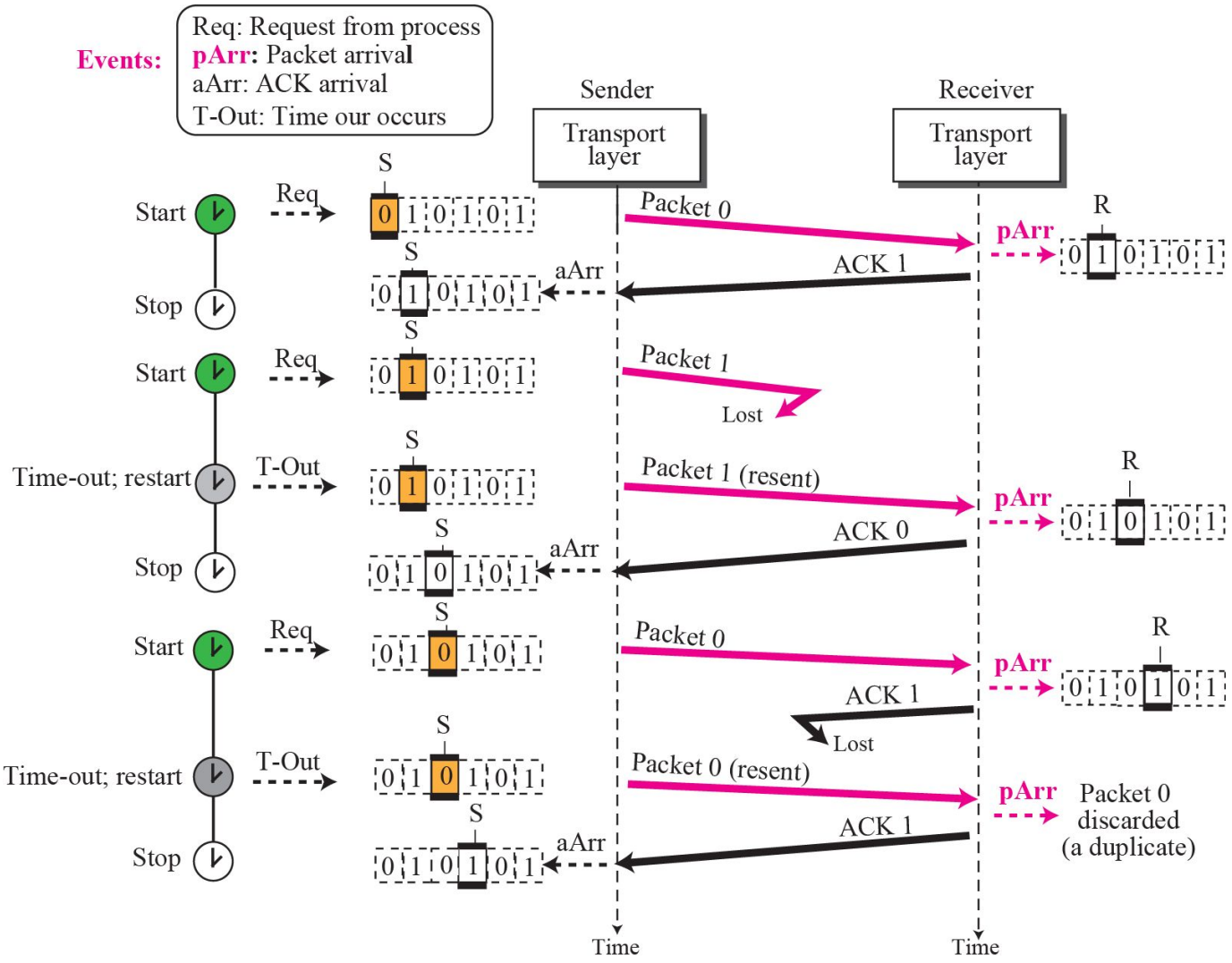


Example

Figure 21 shows an example of Stop-and-Wait protocol.

- **Packet 0 is sent and acknowledged.**
- **Packet 1 is lost and resent after the time-out.**
- **The resent packet 1 is acknowledged and the timer stops.**
- **Packet 0 is sent and acknowledged, but the acknowledgment is lost.**
- **The sender has no idea if the packet or the acknowledgment is lost, so after the time-out, it resends packet 0, which is acknowledged.**

Figure 21 *Example*



Example 5

Assume that, in a Stop-and-Wait system, the bandwidth of the line is 1 Mbps, and 1 bit takes 20 milliseconds to make a round trip. What is the bandwidth-delay product? If the system data packets are 1,000 bits in length, what is the utilization percentage of the link?

Solution

- The bandwidth-delay product is $(1 \times 10^6) \times (20 \times 10^{-3}) = 20{,}000$ bits. The system can send 20,000 bits during the time it takes for the data to go from the sender to the receiver and the acknowledgment to come back. However, the system sends only 1,000 bits. We can say that the link utilization is only 1,000/20,000, or 5 percent. For this reason, for a link with a high bandwidth or long delay, the use of Stop-and-Wait wastes the capacity of the link.

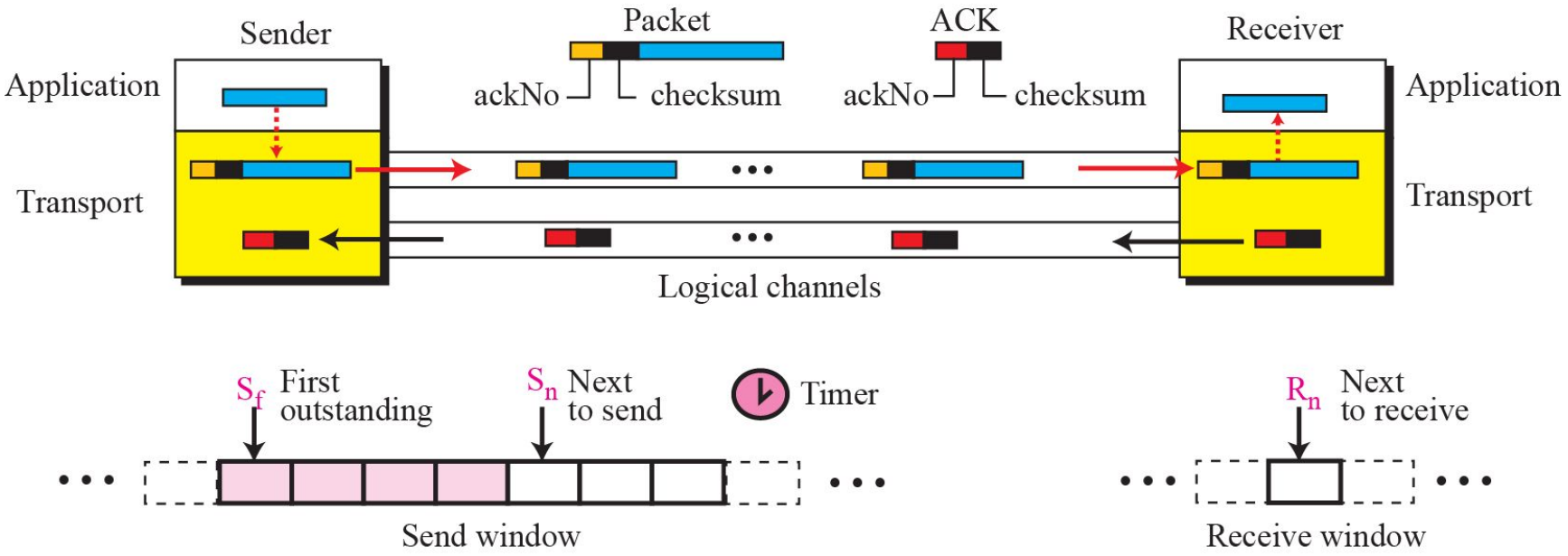
Example 6

What is the utilization percentage of the link in Example 5 if we have a protocol that can send up to 15 packets before stopping and worrying about the acknowledgments?

Solution

- The bandwidth-delay product is still 20,000 bits. The system can send up to 15 packets or 15,000 bits during a round trip. This means the utilization is 15,000/20,000, or 75 percent. Of course, if there are damaged packets, the utilization percentage is much less because packets have to be resent.

Figure 22 *Go-Back-N protocol*



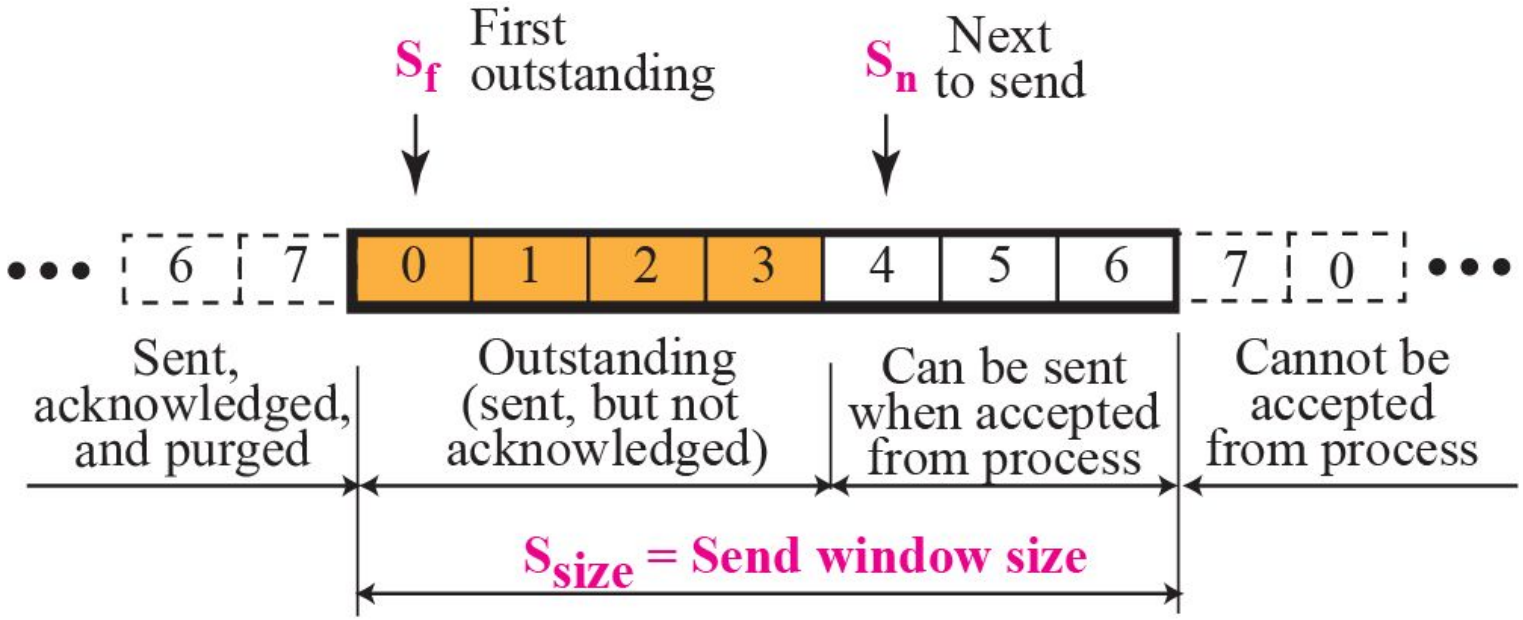
Note

In the Go-Back-N Protocol, the sequence numbers are modulo 2^m, where m is the size of the sequence number field in bits.

Note

In the Go-Back-N protocol, the acknowledgment number is cumulative and defines the sequence number of the next packet expected to arrive.

Figure 23 *Send window for Go-Back-N*



Note

The send window is an abstract concept defining an imaginary box of maximum size = $2^m - 1$ with three variables: S_f, S_n, and S_{size}.

Note

The send window can slide one or more slots when an error-free ACK with ackNo between S_f and S_n (in modular arithmetic) arrives.

Figure 24 *Sliding the send window*

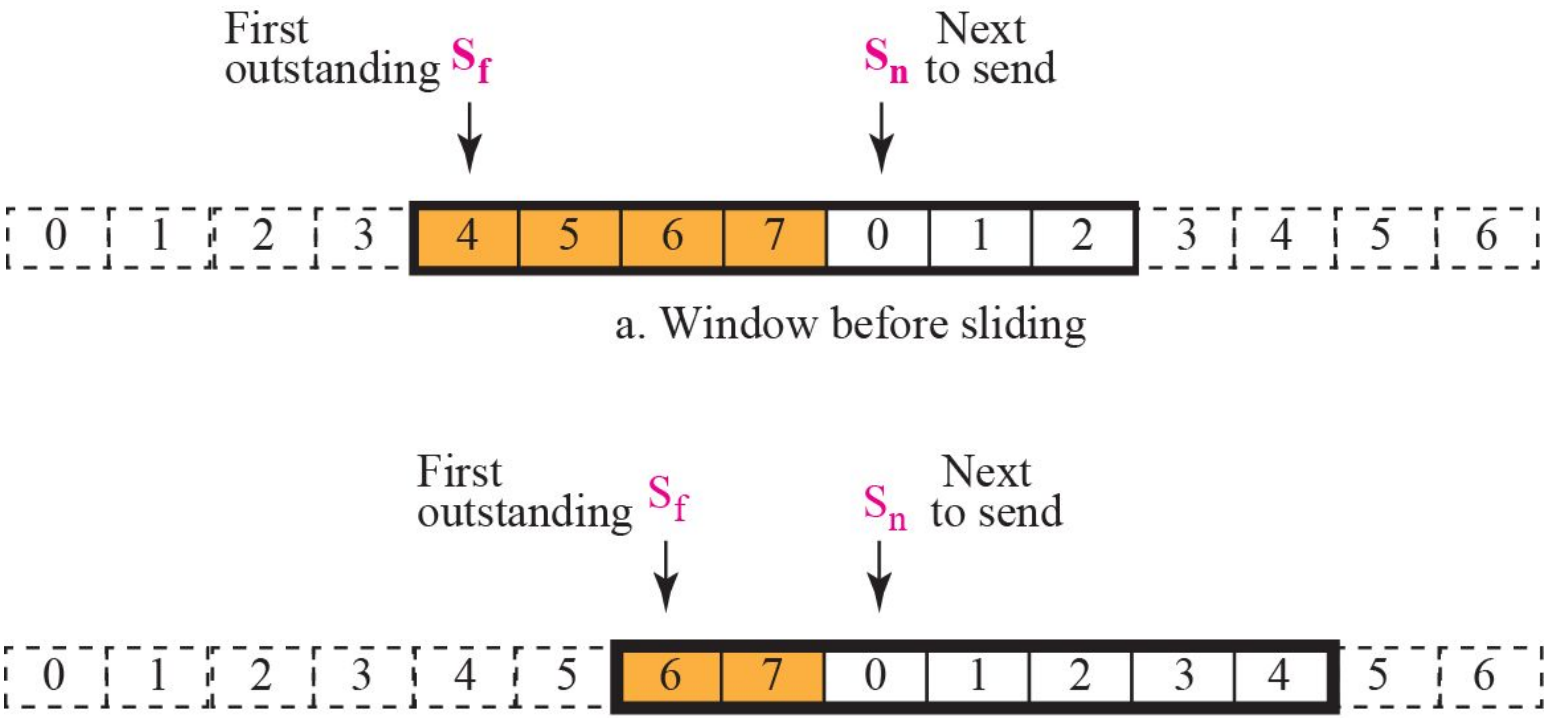
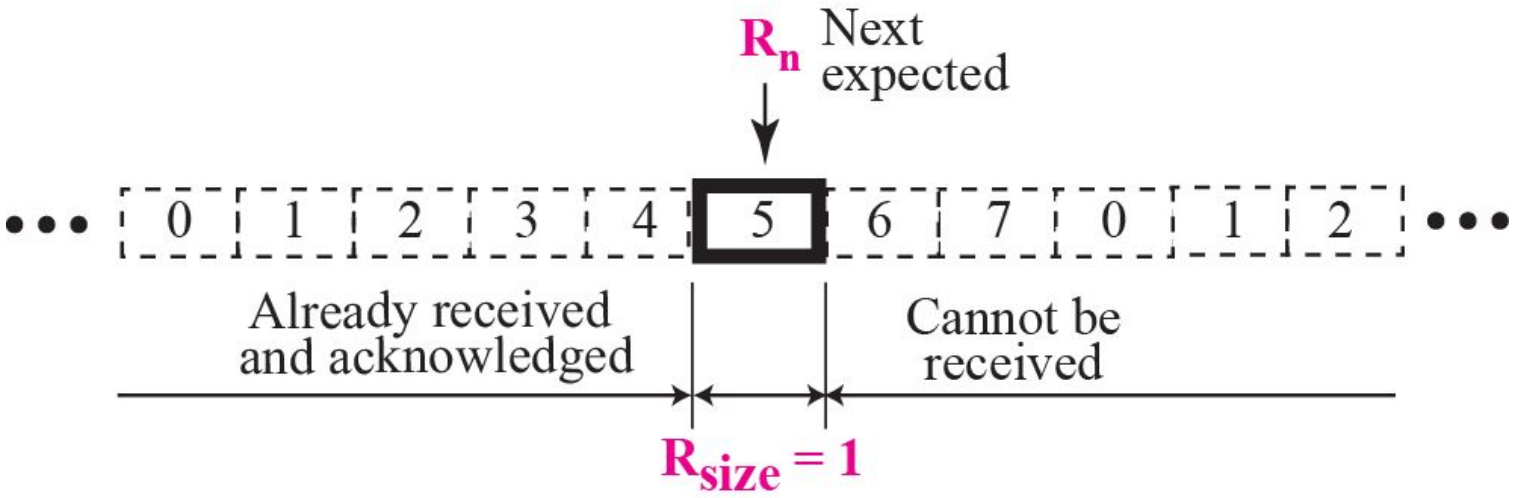


Figure 25 *Receive window for Go-Back-N*



Note

The receive window is an abstract concept defining an imaginary box of size 1 with one single variable R_n.

The window slides when a correct packet has arrived; sliding occurs one slot at a time.

Figure 26 *FSMs for Go-Back-N*

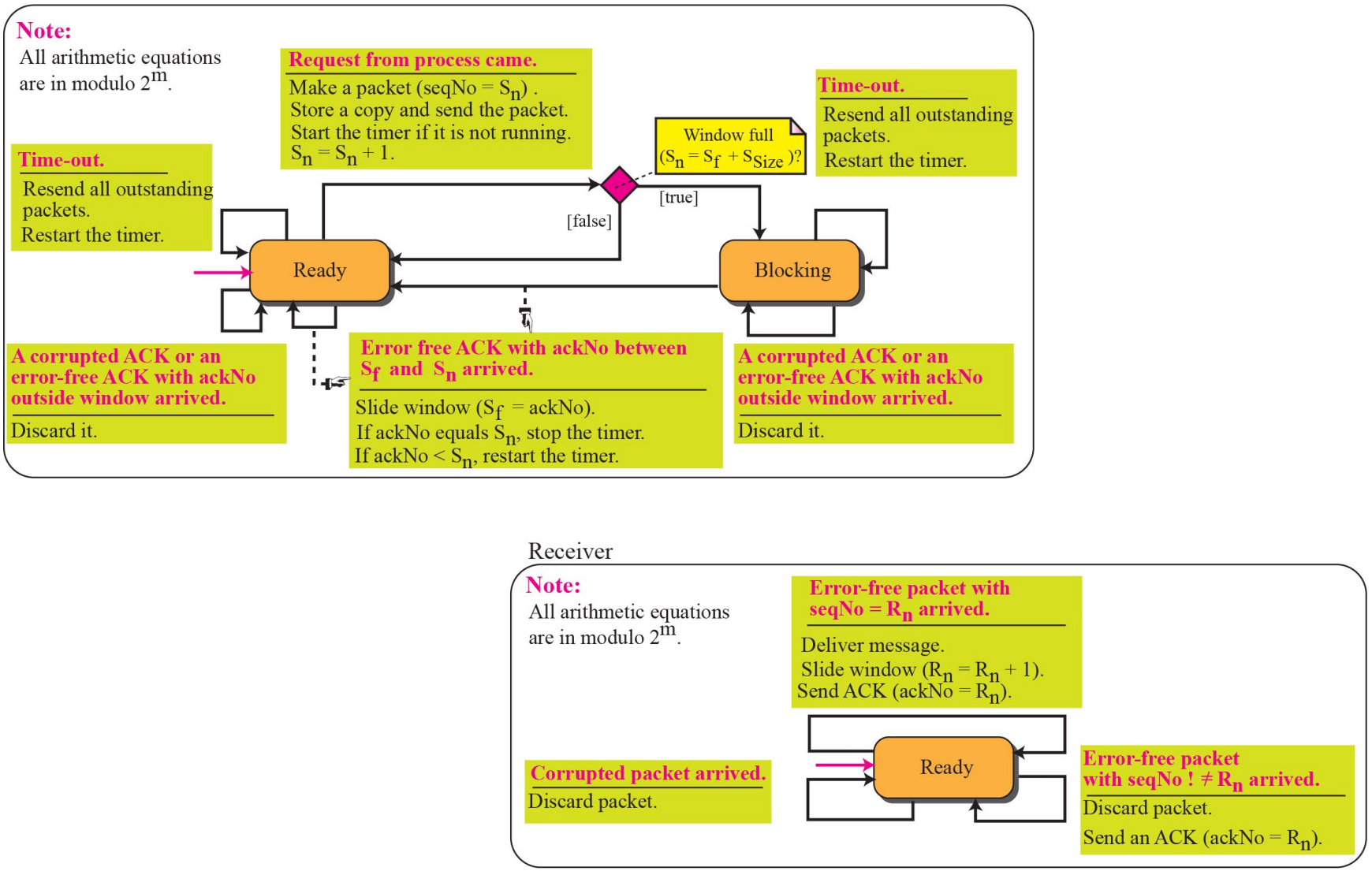


Figure 26 *Go-Back-N: Normal Operation*

- ACK1 is not necessary if ACK2 is sent: Cumulative ACK

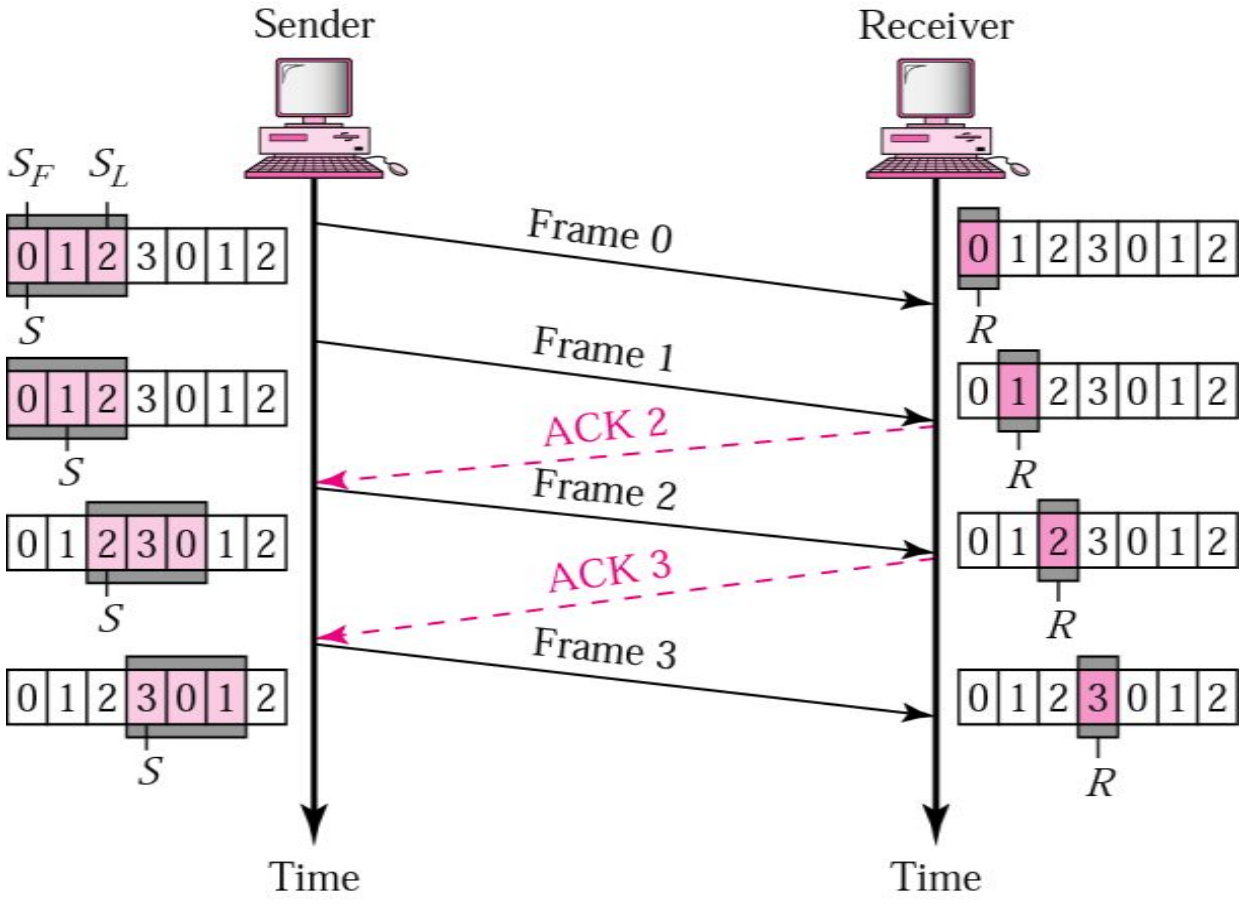


Figure 27 *Go-Back-N : Damaged or Lost Frame*

- Correctly received out of order packets are not Buffered

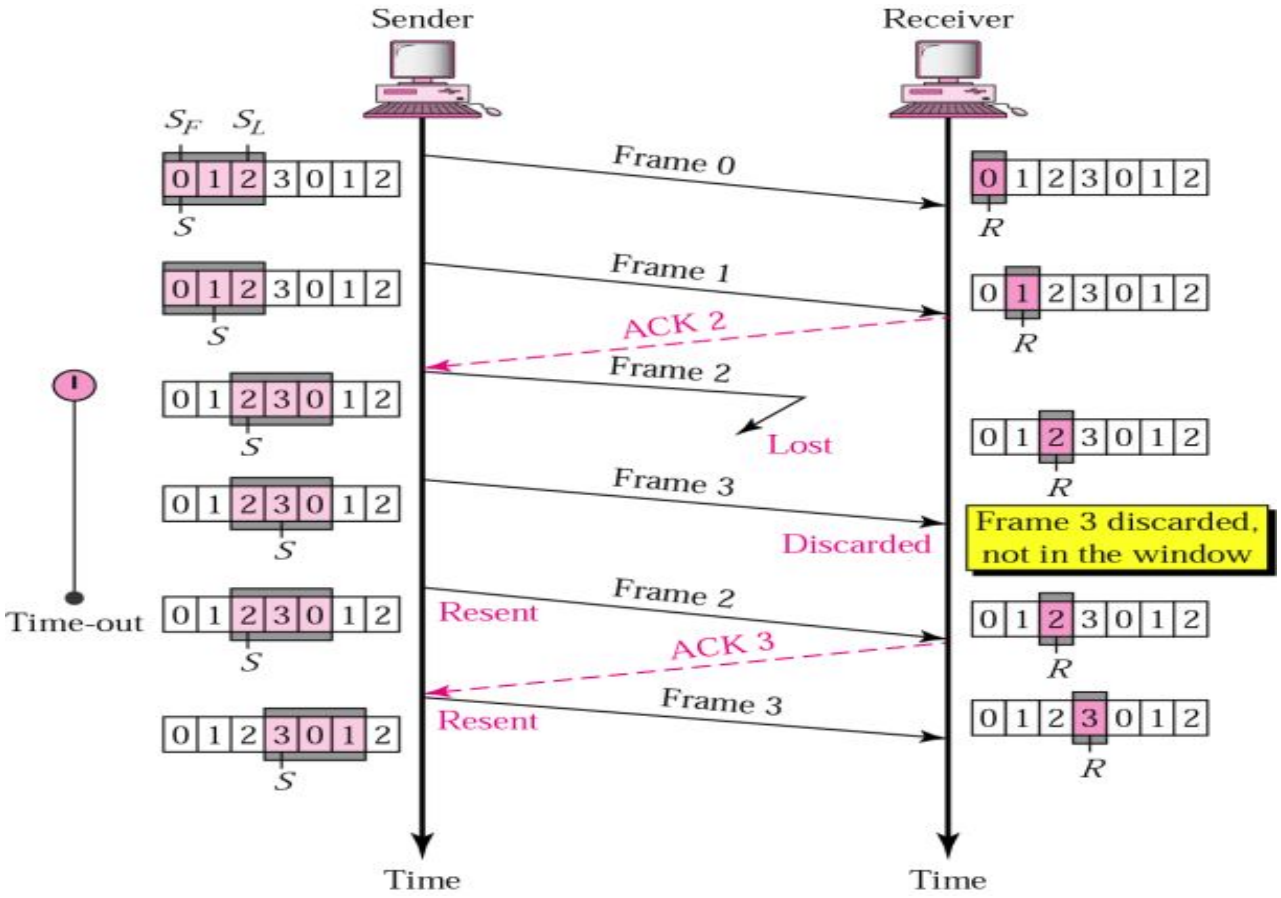
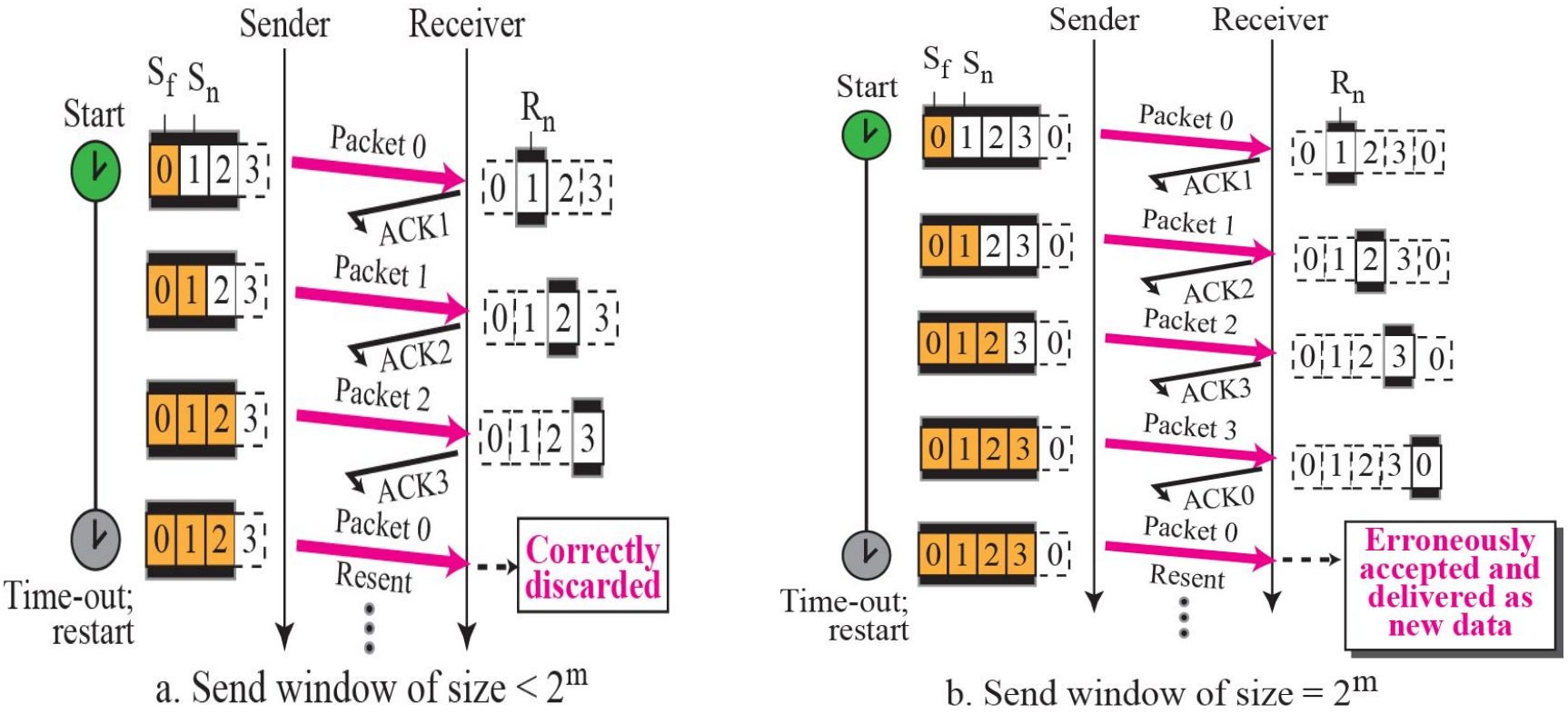


Figure 28 *Send window size for Go-Back-N*



Note

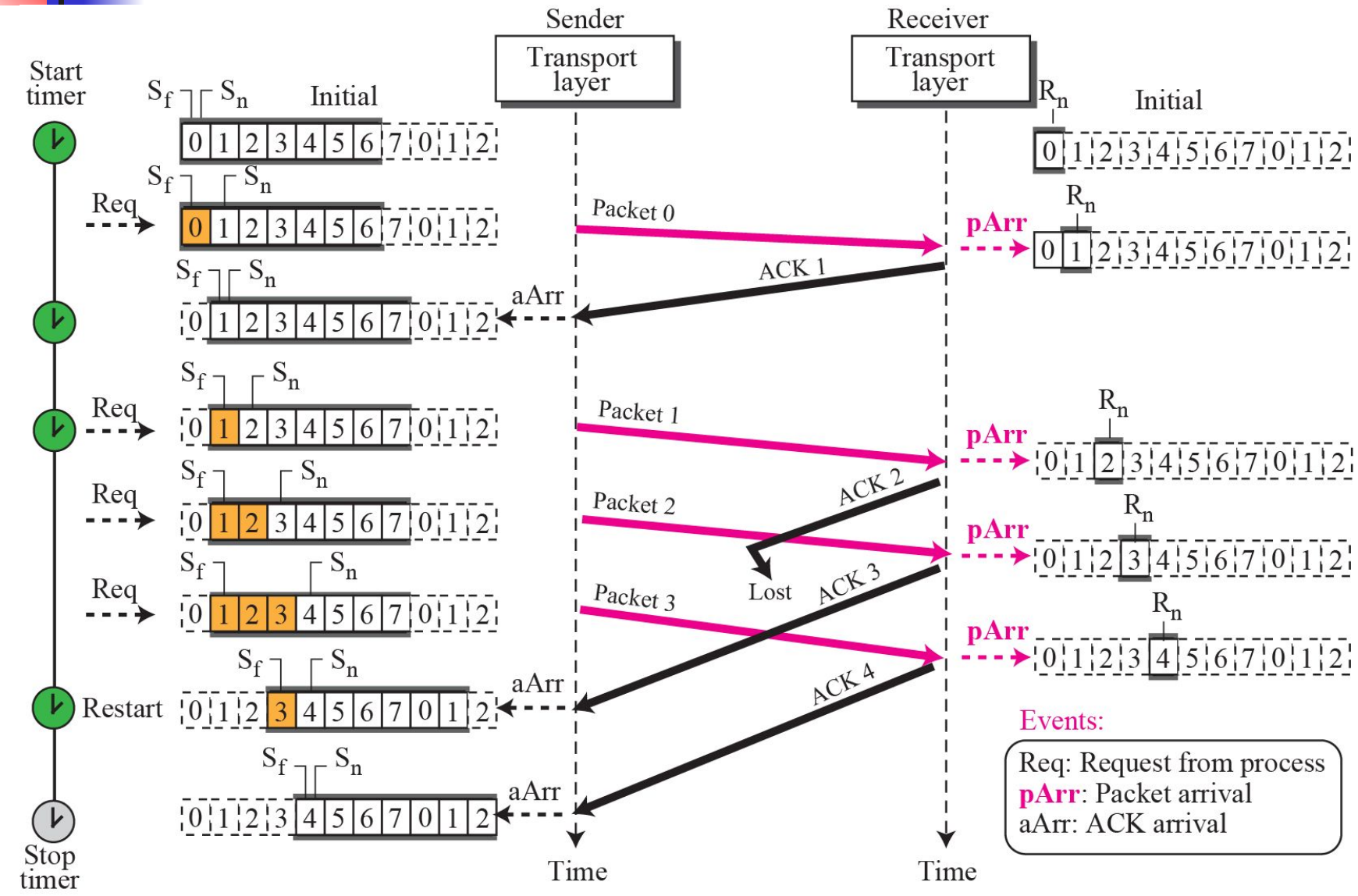
In the Go-Back-N protocol, the size of the send window must be less than 2^m; the size of the receive window is always 1.

Example 7

Figure 28 shows an example of Go-Back-N.

- **This is an example of a case where the forward channel is reliable, but the reverse is not.**
- **No data packets are lost, but some ACKs are delayed and one is lost.**
- **The example also shows how cumulative acknowledgments can help if acknowledgments are delayed or lost.**

Figure 28 *Example*



Example 8

Figure 29 shows what happens when a packet is lost.

- **Packets 0, 1, 2, and 3 are sent. However, packet 1 is lost.**
- **The receiver receives packets 2 and 3, but they are discarded because they are received out of order (packet 1 is expected). When the receiver receives packets 2 and 3, it sends ACK1 to show that it expects to receive packet 1.**
- **However, these ACKs are not useful for the sender because the ackNo is equal S_f, not greater that S_f.**
- **So the sender discards them. When the time-out occurs, the sender resends packets 1, 2, and 3, which are acknowledged.**

Figure 29 *Example*

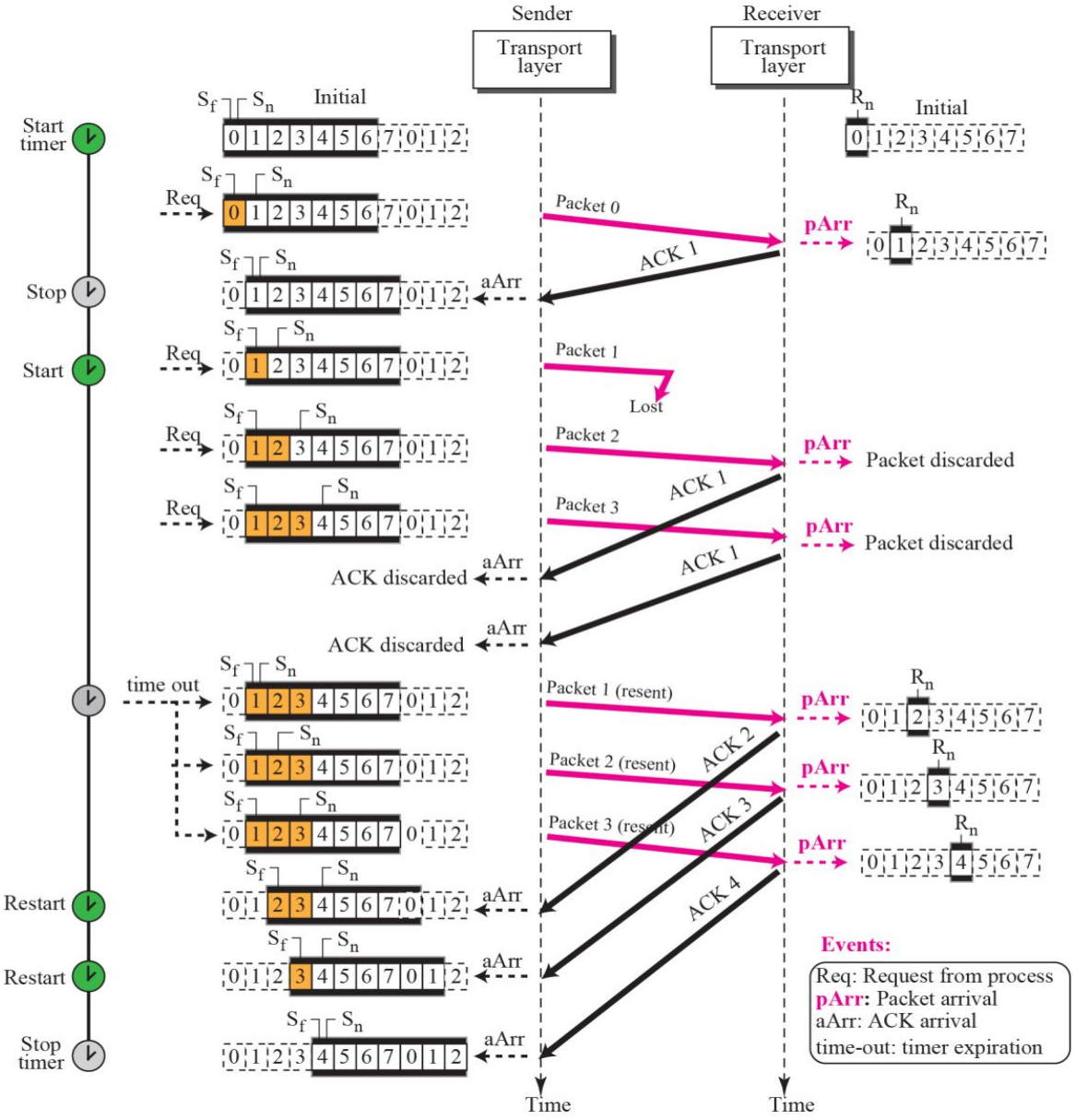


Figure 30 *Outline of Selective-Repeat*

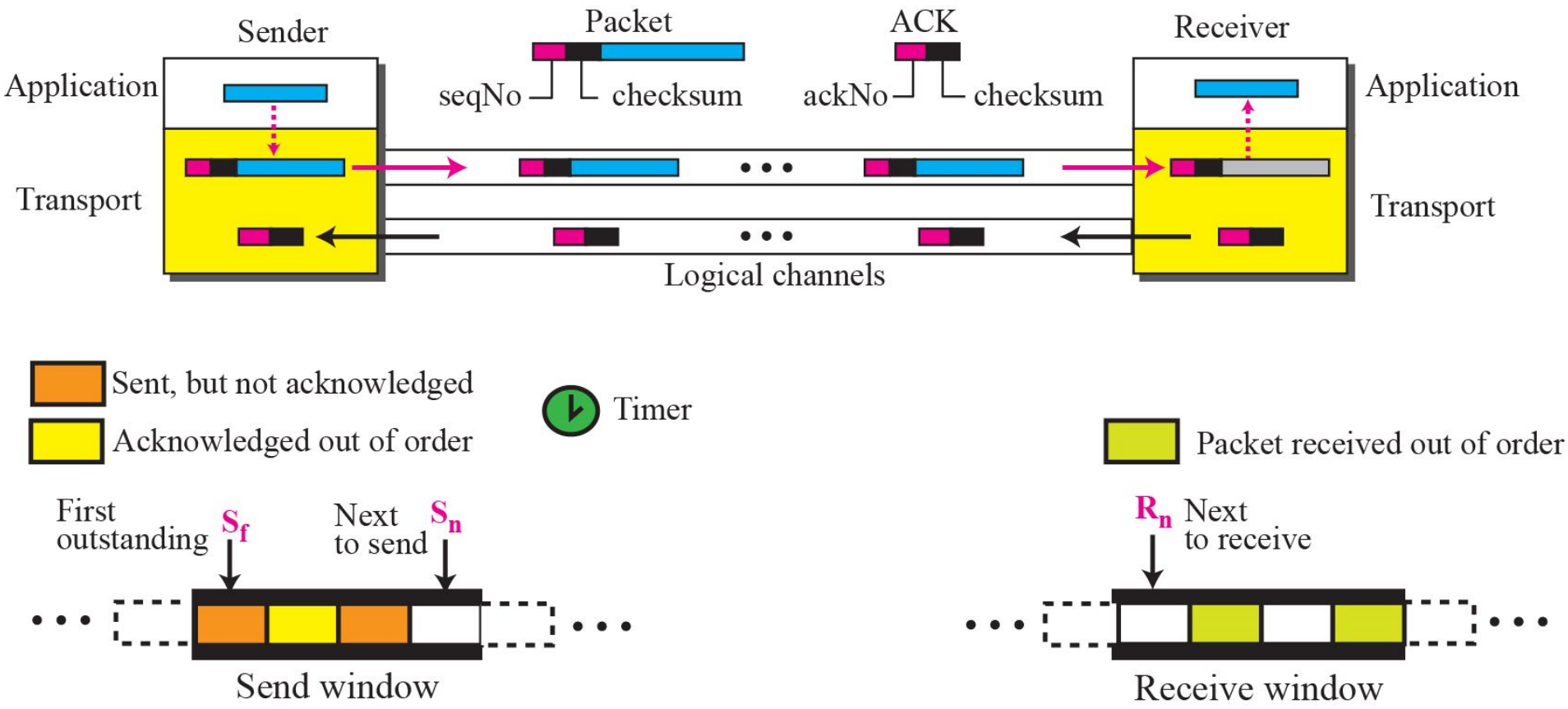


Figure 31 *Send window for Selective-Repeat protocol*

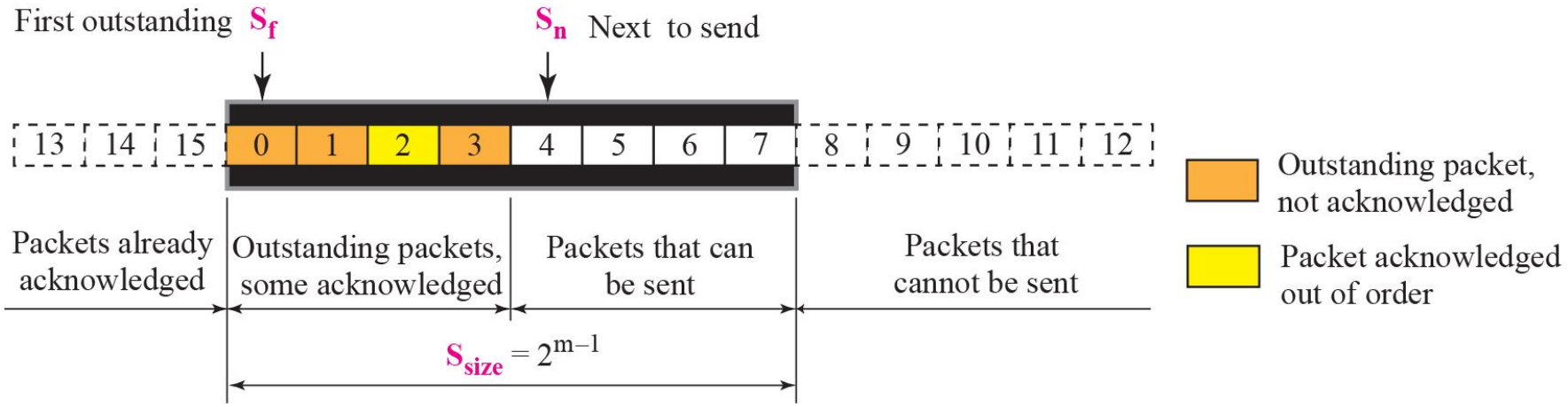
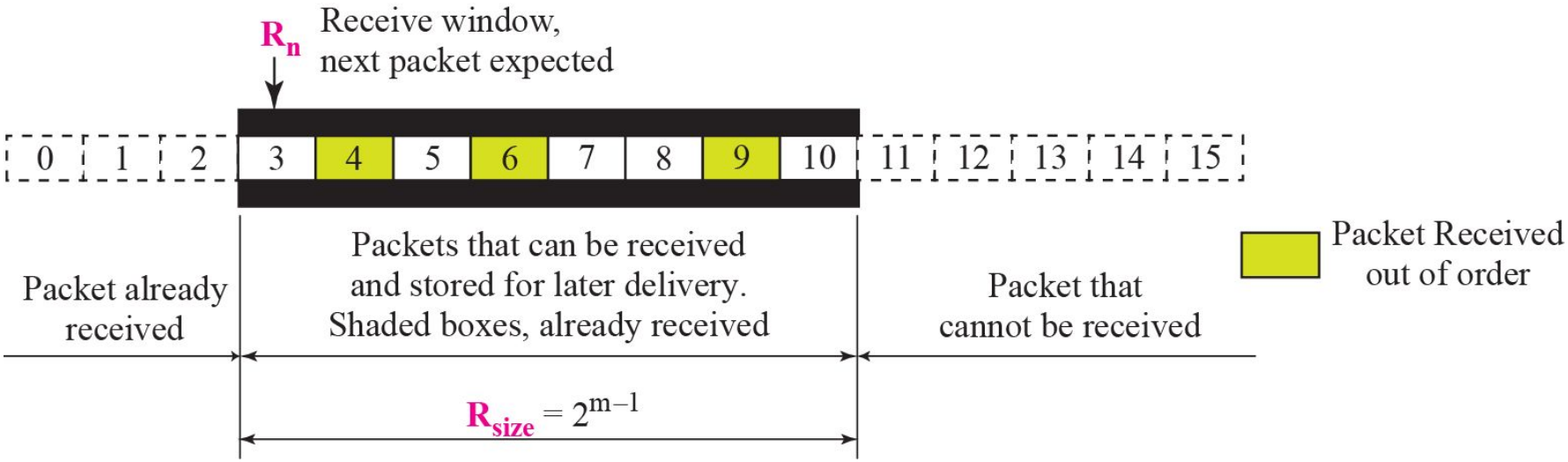


Figure 32 *Receive window for Selective-Repeat protocol*



Note

In the Selective-Repeat protocol, an acknowledgment number defines the sequence number of the error-free packet received.

Example 9

Assume a sender sends 6 packets: packets 0, 1, 2, 3, 4, and 5. The sender receives an ACK with ackNo = 3. What is the interpretation if the system is using GBN or SR?

Solution

If the system is using GBN, it means that packets 0, 1, and 2 have been received uncorrupted and the receiver is expecting packet 3. If the system is using SR, it means that packet 3 has been received uncorrupted; the ACK does not say anything about other packets.

Example 10

This example is similar to Example 8 (Figure 29) in which packet 1 is lost. We show how Selective-Repeat behaves in this case.

Figure 34 shows the situation.
At the sender, packet 0 is transmitted and acknowledged. Packet 1 is lost.
Packets 2 and 3 arrive out of order and are acknowledged. When the timer times out, packet 1 (the only unacknowledged packet) is resent and is acknowledged.
The send window then slides.

Figure 34 *Example*

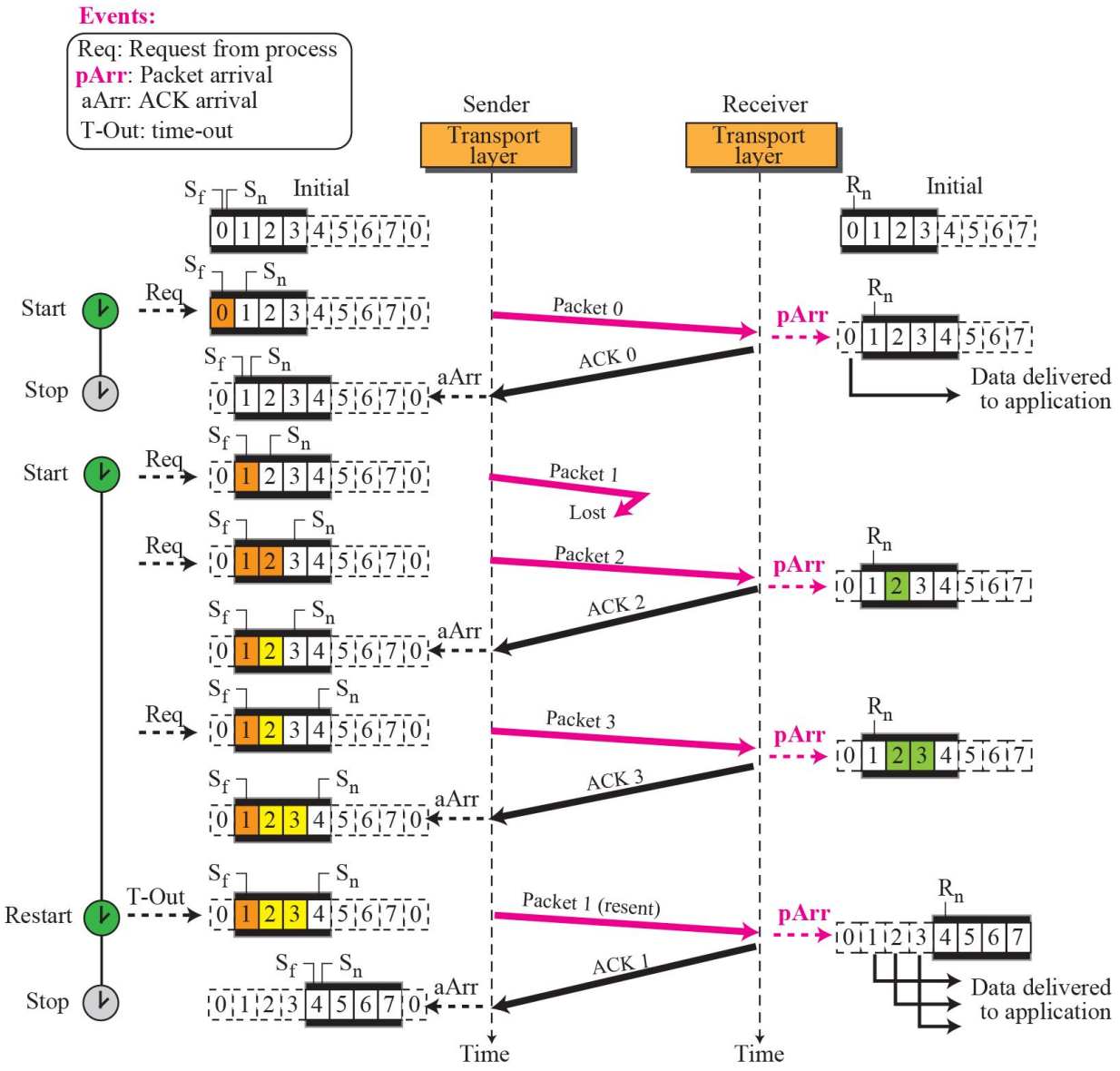
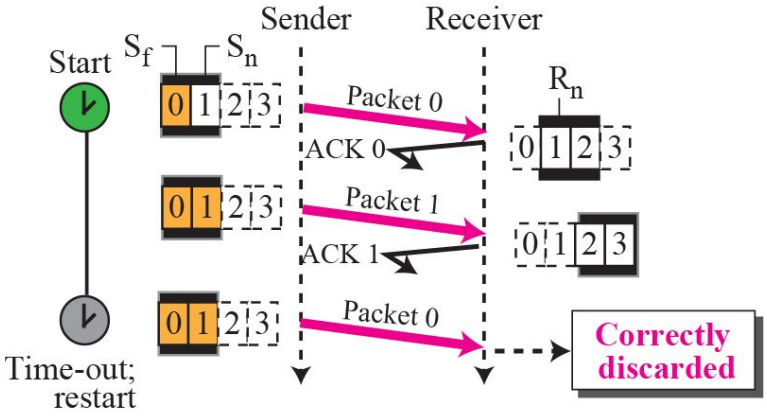
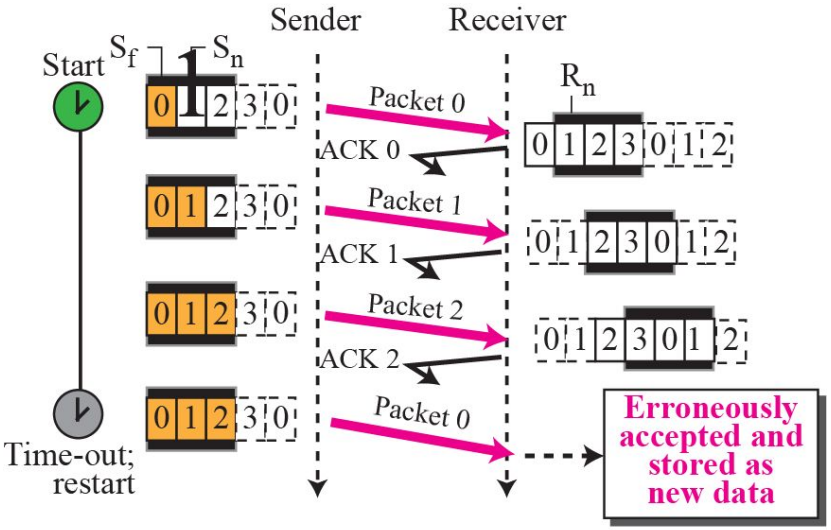


Figure 35 *Selective-Repeat window size*



a. Send and receive windows of size = 2^{m-1}



b. Send and receive windows of size > 2^{m-1}

Note

In Selective-Repeat, the size of the sender and receiver window can be at most one-half of 2m.

Figure 36 *Design of piggybacking for Go-Back-N*

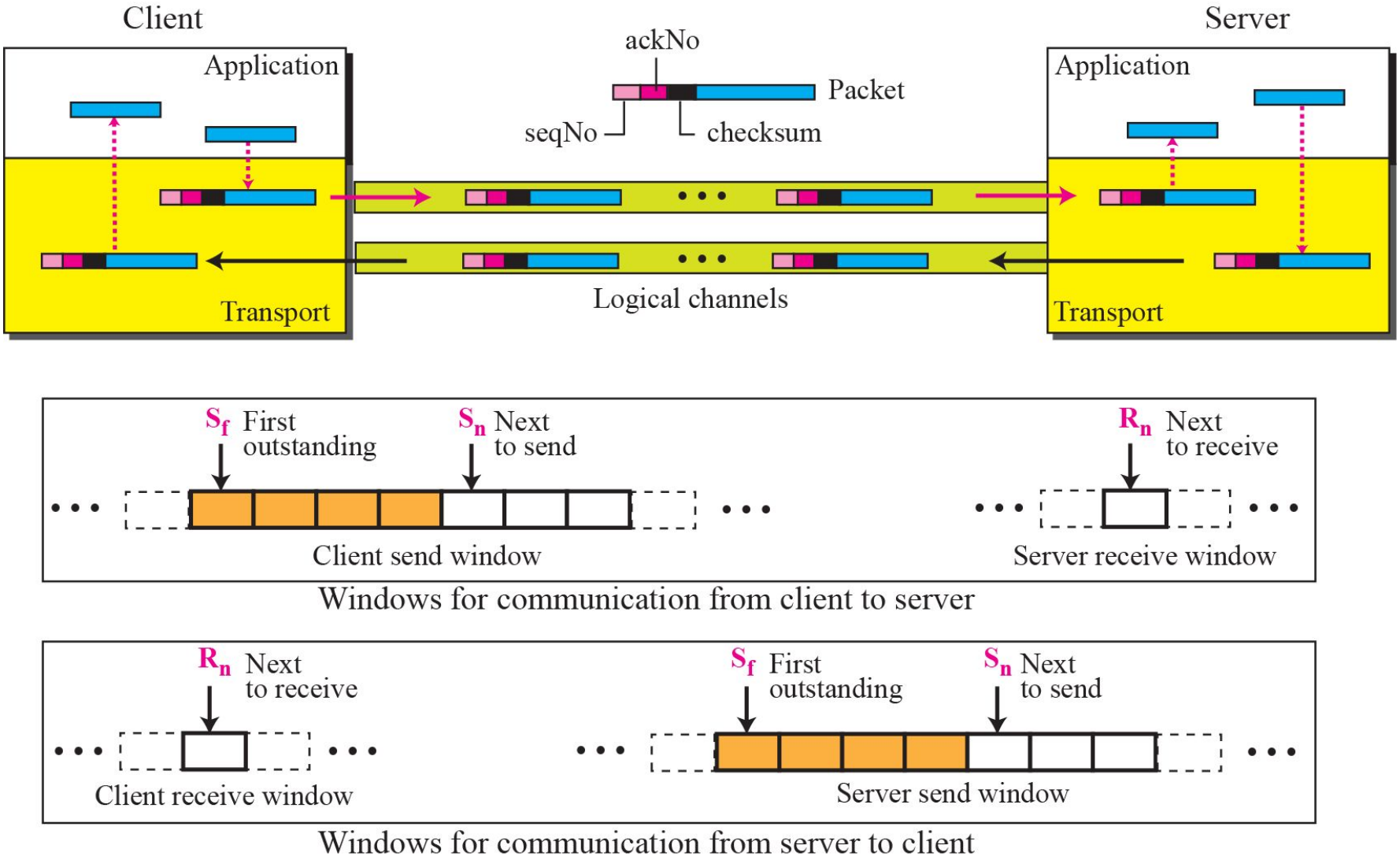


Figure 29 Piggybacking (Bidirectional transmission)

- It is a method to combine a data frame with an acknowledgment.
- It can save bandwidth because data frame and an ACK frame can combined into just one frame