Computer Network

Lecture-20

Dharmendra Kumar (Associate Professor)
Department of Computer Science and Engineering
United College of Engineering and Research,
Prayagraj

Data Link Control

Data Link Control

Framing:

- Framing in the data link layer separates a message from one source to a destination, or from other messages to other destinations, by adding a sender address and a destination address. The destination address defines where the packet is to go; the sender address helps the recipient acknowledge the receipt.
- Frames can be of fixed size or of variable size.

Fixed-Size Framing

In fixed-size framing, there is no need for defining the boundaries of the frames; the size itself can be used as a delimiter. An example of this type of framing is the ATM wide-area network, which uses frames of fixed size called cells.

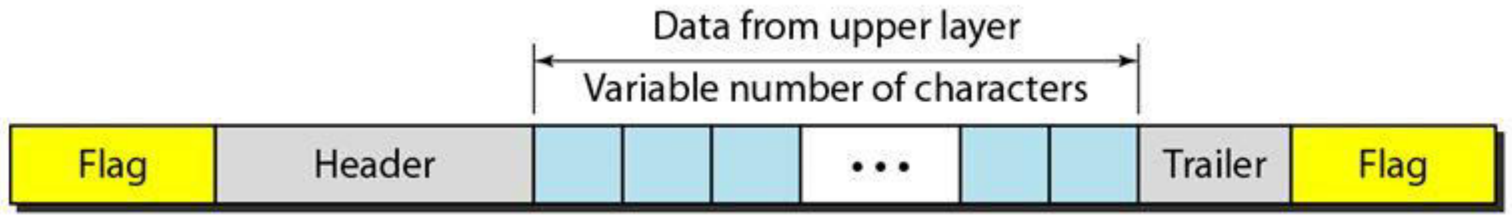
Data Link Control

Variable-Size Framing

In variable-size framing, we need a way to define the end of the frame and the beginning of the next. Historically, two approaches were used for this purpose: a character-oriented approach and a bit-oriented approach.

Character-Oriented Protocols (byte oriented)

In a character-oriented protocol, data to be carried are 8-bit characters from a coding system such as ASCII. The header, which normally carries the source and destination addresses and other control information, and the trailer, which carries error detection or error correction redundant bits, are also multiples of 8 bits. To separate one frame from the next, an 8-bit (1-byte) flag is added at the beginning and the end of a frame. The flag, composed of protocol-dependent special characters, signals the start or end of a frame. Following figure shows the format of a frame in a character-oriented protocol.



Character-Oriented Protocols (byte oriented)

Any pattern used for the flag could also be part of the information. If this happens, the receiver, when it encounters this pattern in the middle of the data, thinks it has reached the end of the frame. To fix this problem, a byte-stuffing strategy was added to character-oriented framing. In byte stuffing (or character stuffing), a special byte is added to the data section of the frame when there is a character with the same pattern as the flag. The data section is stuffed with an extra byte. This byte is usually called the escape character (ESC), which has a predefined bit pattern. Whenever the receiver encounters the ESC character, it removes it from the data section and treats the next character as data, not a delimiting flag.

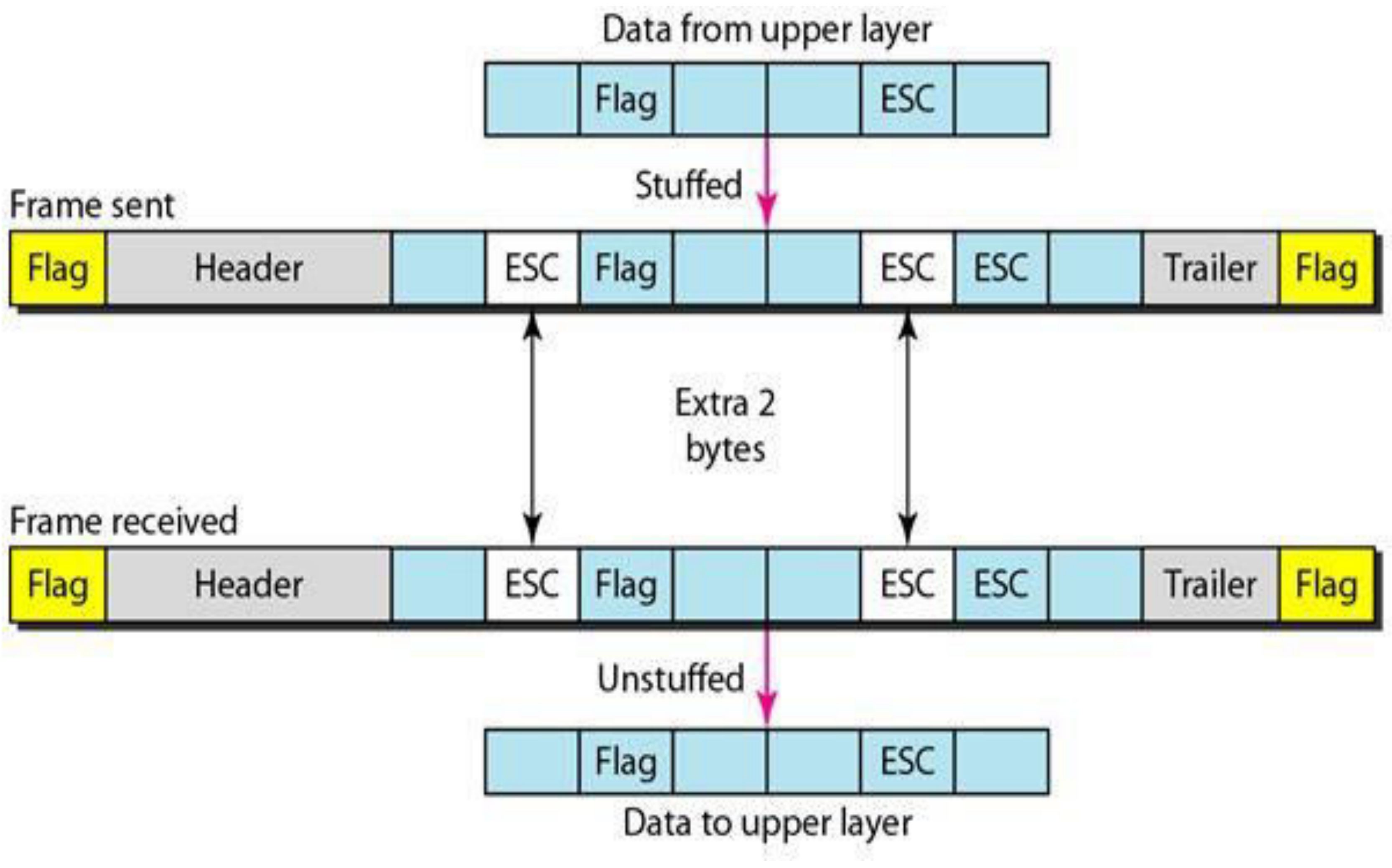
Character-Oriented Protocols (byte oriented)

Byte stuffing by the escape character allows the presence of the flag in the data section of the frame, but it creates another problem. What happens if the text contains one or more escape characters followed by a flag? The receiver removes the escape character, but keeps the flag, which is incorrectly interpreted as the end of the frame. To solve this problem, the escape characters that are part of the text must also be marked by another escape character. In other words, if the escape character is part of the text, an extra one is added to show that the second one is part of the text.

Note: Byte stuffing is the process of adding 1 extra byte whenever there is a flag or escape character in the text.

Character-Oriented Protocols (byte oriented)

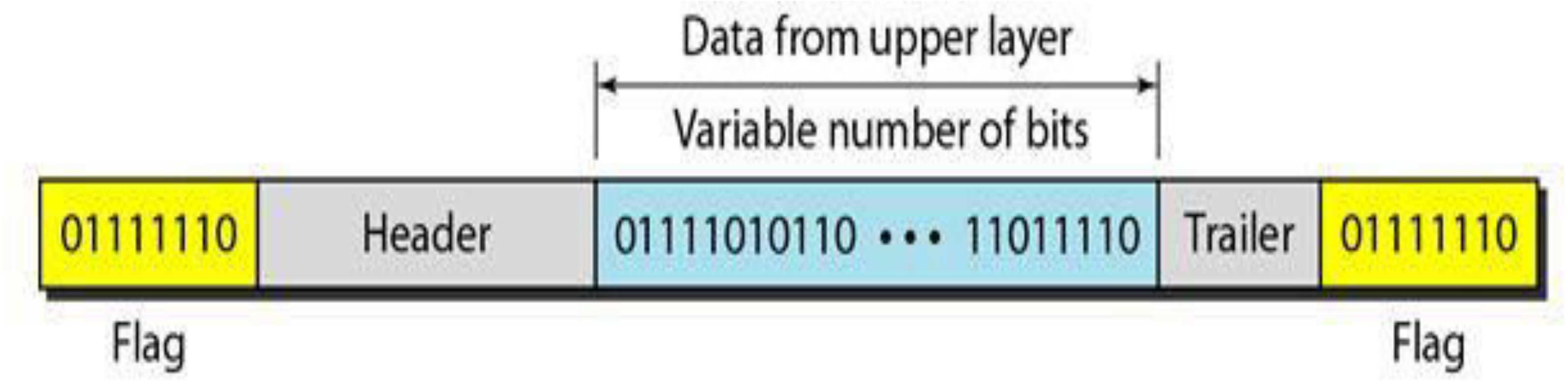
Byte stuffing and unstuffing



Bit-Oriented Protocols

Bit-Oriented Protocols

In a bit-oriented protocol, the data section of a frame is a sequence of bits to be interpreted by the upper layer as text, graphic, audio, video, and so on. However, in addition to headers (and possible trailers), we still need a delimiter to separate one frame from the other. Most protocols use a special 8-bit pattern flag 01111110 as the delimiter to define the beginning and the end of the frame, as shown in following figure .

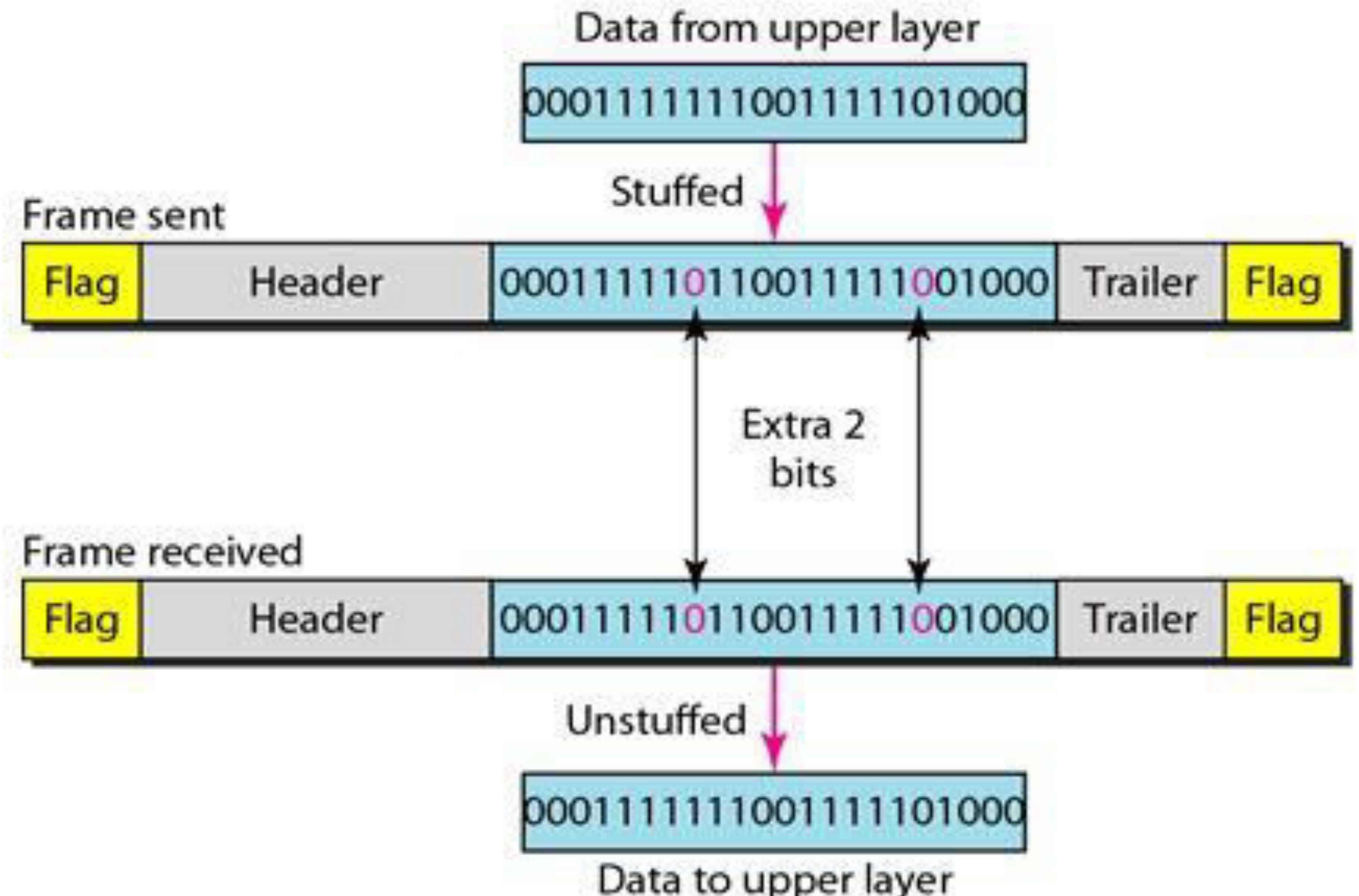


Bit-Oriented Protocols

This flag can create the same type of problem we saw in the byte-oriented protocols. That is, if the flag pattern appears in the data, we need to somehow inform the receiver that this is not the end of the frame. We do this by stuffing 1 single bit (instead of 1 byte) to prevent the pattern from looking like a flag. The strategy is called bit stuffing. In bit stuffing, if a 0 and five consecutive 1 bits are encountered, an extra 0 is added. This extra stuffed bit is eventually removed from the data by the receiver. Note that the extra bit is added after one 0 followed by five 1's regardless of the value of the next bit. This guarantees that the flag field sequence does not inadvertently appear in the frame.

Note: Bit stuffing is the process of adding one extra 0 whenever five consecutive 1's follow a 0 in the data, so that the receiver does not mistake the pattern 0111110 for a flag.

Bit-Oriented Protocols



Bit-Oriented Protocols

Figure shows bit stuffing at the sender and bit removal at the receiver. Note that even if we have a 0 after five 1's, we still stuff a 0. The 0 will be removed by the receiver. If the flag like pattern 01111110 appears in the data, it will change to 011111010 (stuffed) and is not mistaken as a flag by the receiver. The real flag 01111110 is not stuffed by the sender and is recognized by the receiver.

Flow and Error Control

The most important responsibilities of the data link layer are flow control and error control.

Flow control

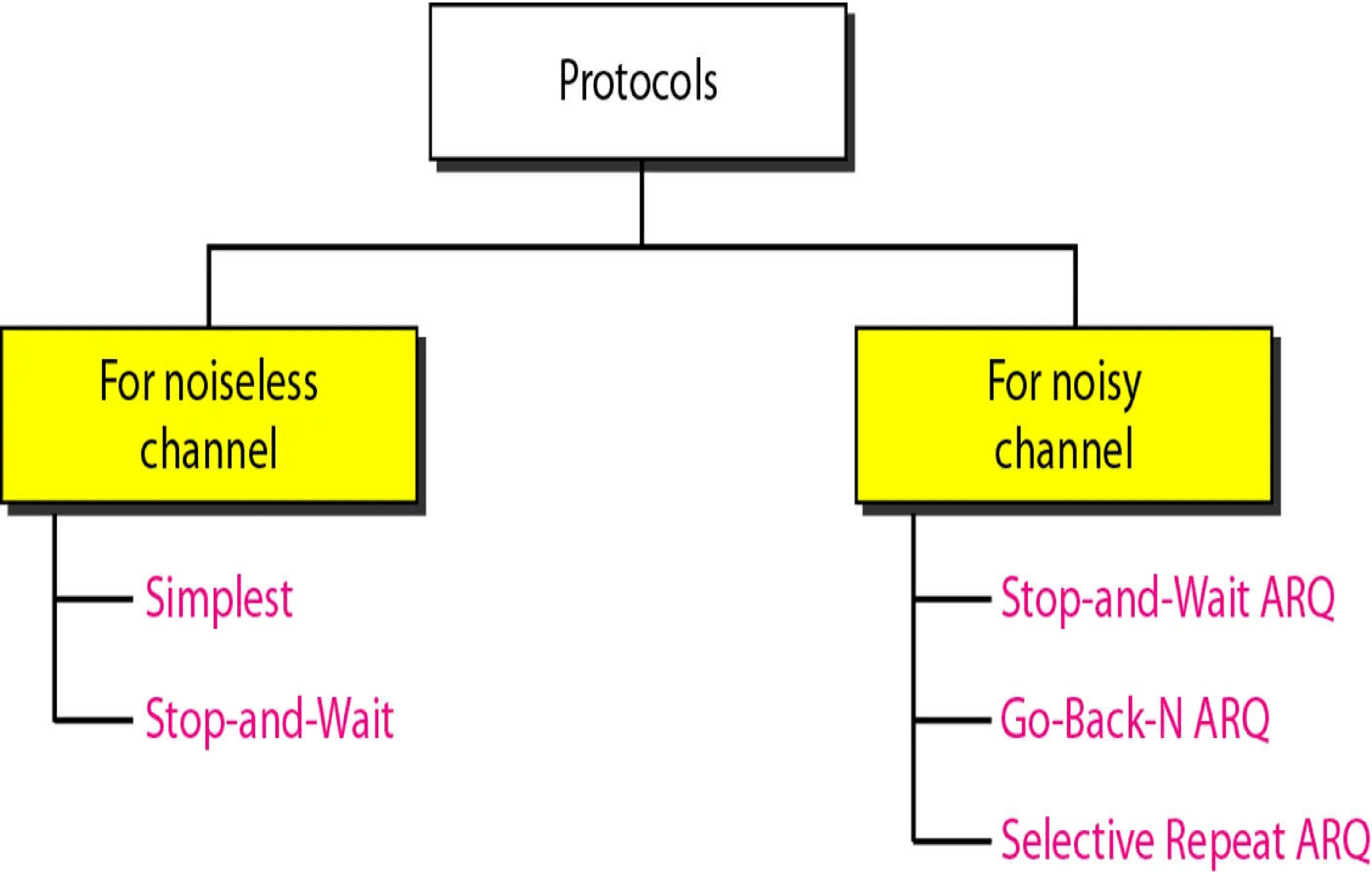
- Flow control coordinates the amount of data that can be sent before receiving an acknowledgment.
- In most protocols, flow control is a set of procedures that tells the sender how much data it can transmit before it must wait for an acknowledgment from the receiver.
- The flow of data must not be allowed to overwhelm the receiver. Any receiving device has a limited speed at which it can process incoming data and a limited amount of memory in which to store incoming data. The receiving device must be able to inform the sending device before those limits are reached and to request that the transmitting device send fewer frames or stop temporarily.

Flow and Error Control

Error control

- Error control is both error detection and error correction. It allows the receiver to inform the sender of any frames lost or damaged in transmission and coordinates the retransmission of those frames by the sender.
- Error control in the data link layer is often implemented simply: Any time an error is detected in an exchange, specified frames are retransmitted. This process is called automatic repeat request (ARQ).

Data link control protocol



Data link control protocol

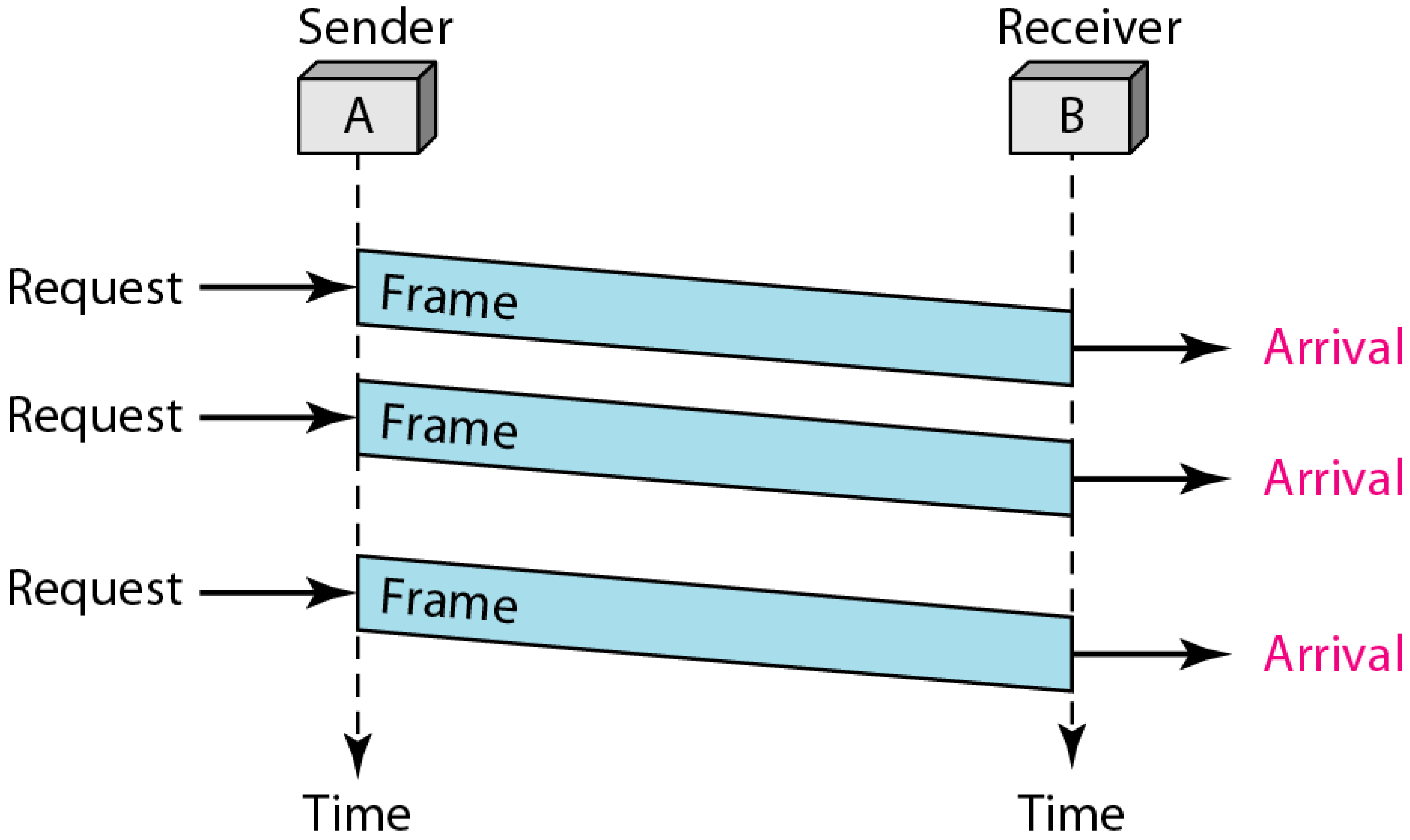
Noiseless Channels

- Assume we have an ideal channel in which no frames are lost, duplicated, or corrupted.
- We have two protocols for this type of channel. The first is a protocol that does not use flow control; the second is the one that does.
- Neither has error control because we have assumed that the channel is a perfect noiseless channel.

Simplest Protocol

- It is one that has no flow or error control.
- It is a unidirectional protocol in which data frames are traveling in only one direction-from the sender to receiver.
- We assume that the receiver can immediately handle any frame it receives with a processing time that is small enough to be negligible. The data link layer of the receiver immediately removes the header from the frame and hands the data packet to its network layer, which can also accept the packet immediately. In other words, the receiver can never be overwhelmed with incoming frames.

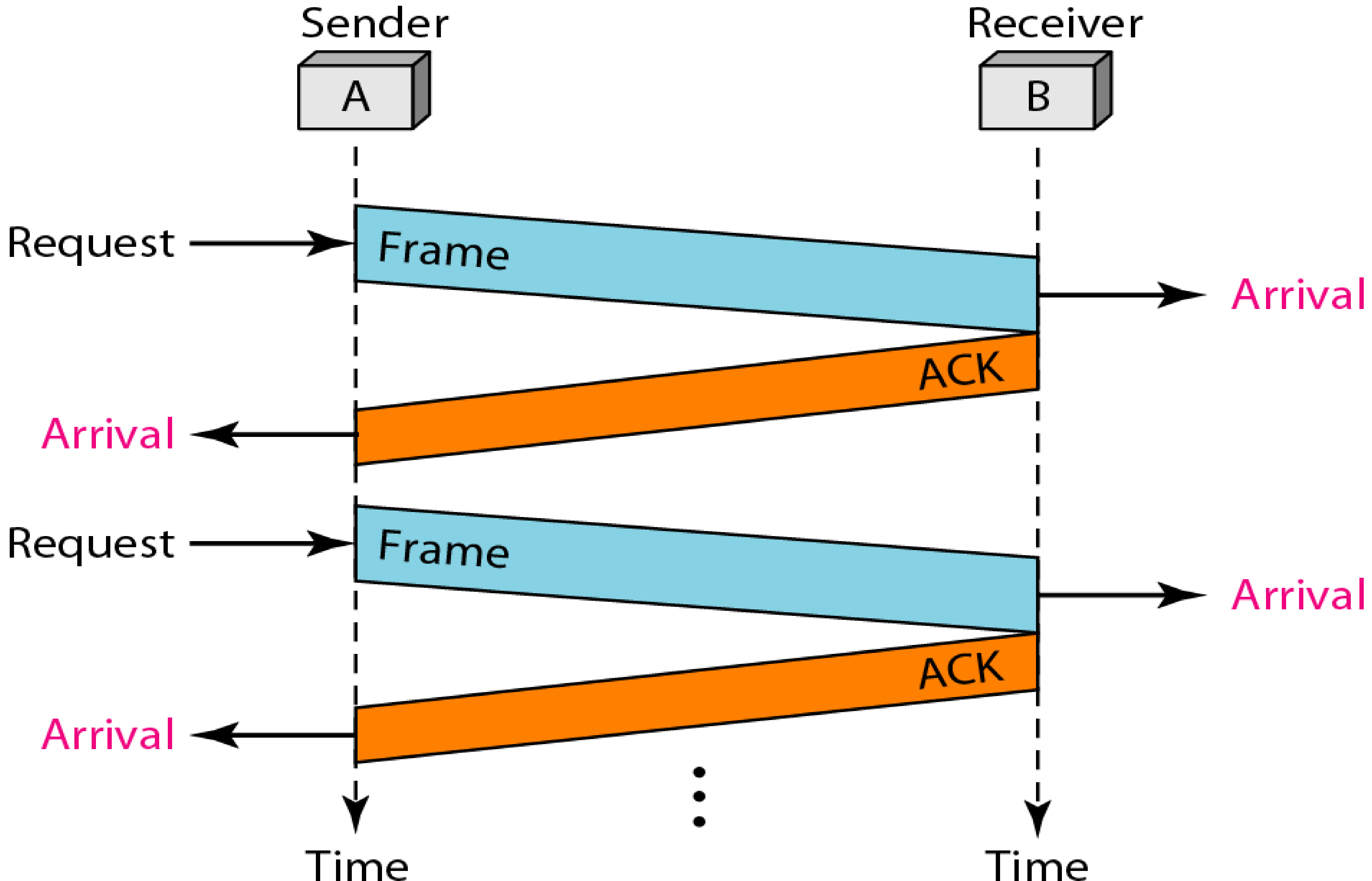
Simplest Protocol



Stop-and-Wait Protocol

- The sender sends one frame, stops until it receives confirmation from the receiver (okay to go ahead), and then sends the next frame.
- We still have unidirectional communication for data frames, but auxiliary ACK frames (simple tokens of acknowledgment) travel from the other direction.
- We add flow control to our previous protocol.

Stop-and-Wait Protocol


Sender
Receiver
A
B
Request
Frame
Arrival
ACK
Arrival
Request
Frame
Arrival
ACK
Arrival
Time
Time