1 The Data Link Layer

1.1 Data Link layer functions and services

1.1.1 Main functions

- Provide service interface to the network layer.
- Eliminate/reduce transmission errors.
- Regulate data flow: Slow receivers not swamped by fast senders.

1.1.2 Services provided

Principal service: Transfer data from the network layer on the source machine to the network layer on the destination machine.

The actual services that are offered vary from protocol to protocol. Three reasonable possibilities that we will consider in turn are:

- **Unacknowledged connectionless service:**
  - No logical connection is established beforehand or released
  - Source machine sends independent frames without having the destination machine acknowledge them. afterwards.
  - If a frame is lost due to noise on the line, no attempt is made to detector recover from that loss.
  - Appropriate when the error rate is very low and for real-time traffic.
- **Acknowledged connectionless service:**
  - No logical connections used.
  - Each frame sent is individually acknowledged so that the sender knows whether a frame has arrived correctly or been lost.
  - If it has not arrived within a specified time interval, it can be sent again.
  - This service is useful over unreliable channels, such as wireless systems.(i.e. Wi-Fi).
- **Acknowledged connection-oriented service:**
  - The source and destination machines establish a connection before any data are transferred.
  - Each frame is numbered, and the data link layer guarantees that each frame sent is indeed received.
  - it guarantees that each frame is received exactly once and that all frames are received in the right order.

  - Appropriate over long, unreliable links (i.e. satellite channel, long-distance telephone circuit). If acknowledged connectionless service were used, lost acknowledgements could cause a frame to be sent and received several times.
  - Divided in 3 phases:
    * **First phase:** The connection is established by having both sides initialize variables and counters needed to keep track of which frames have been received and which ones have not.
    * **Second phase:** One or more frames are actually transmitted.
    * **Third phase:** The connection is released, freeing up the variables, buffers, and other resources used to maintain the connection.

Providing acknowledgements in the data link layer is just an optimization, never a requirement. The network layer can always send a packet and wait for it to be acknowledged by its peer on the remote machine. This strategy,however, can be inefficient. Links usually have a strict maximum frame length imposed by the hardware, and known propagation delays. The network layer does not know these parameters. It might send a large packet that is broken up into 10 frames, of which 2 are lost on average. If individual frames are acknowledged and retransmitted, then errors can be corrected more directly and more quickly. On reliable channels, such as fiber, the overhead of a heavyweight data link protocol may be unnecessary, but on (inherently unreliable) wireless channels it is well worth the cost.

1.2 Framing

The physical layer accepts and sends a raw bit stream. If the channel is noisy the physical layer will add some redundancy to its signals to reduce the bit error rate to a tolerable level. However, the bit stream received by the data link layer is not guaranteed to be error free. It is up to the data link layer to detect and correct errors.

The approach is for the data link layer to break up the bit stream into discrete frames, compute a short token called a checksum for each frame, and include the checksum in the frame when it is transmitted. When a frame arrives at the destination,the checksum is recomputed. If it is different from the one contained in the frame, the data link layer knows that an error has occurred and takes steps to deal with it.

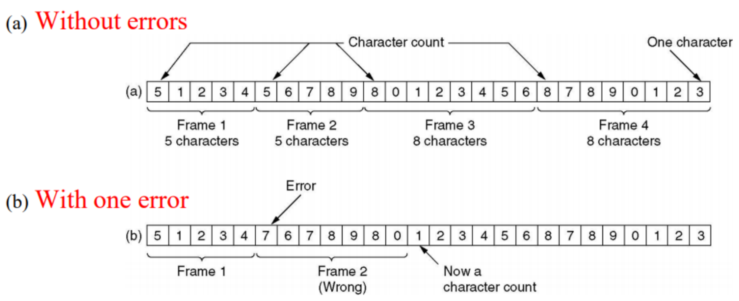
A good design must make it easy for a receiver to find the start of new frames while using little of the channel bandwidth. We will look at three methods:

1.2.1 Byte count

Uses a field in the header to specify the number of bytes in the frame. When the data link layer at the destination sees the byte count, it knows how many bytes follow and hence where the end of the frame is.

Issues

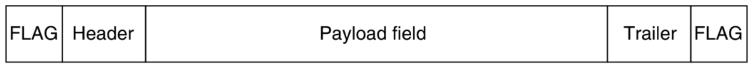
- The count can be garbled by a transmission error.
- A single bit flip, may trigger the destination to get out of synchronization.
- If an out-of-sync occurs it is unable to locate the correct start of the next frame.



This method is rarely used.

1.2.2 Flag bytes with byte stuffing

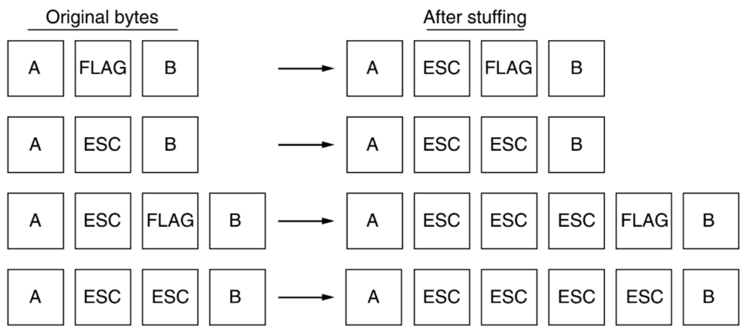
This method gets around the problem of resynchronization by having each frame start and end with special bytes (flag bytes).



However, this flag may occur in the middle of the data and induce the receiver in error by thinking the end of the frame was reached. This issue can be solved using **byte stuffing**.

Byte Stuffing

- Inserting a special escape byte (ESC) before each flag byte in the data.
- Makes framing flag bytes distinguishable from the ones in the data.
- Escape bytes present in the data also need to be escaped.



1.2.3 Flag bits with bit stuffing

- Each frame begins and ends with a special bit pattern (01111110 / 0x7E).
- When the sender finds five consecutive 1 bits in the data, it stuffs a 0 bit into the outgoing bit stream.
- When the receiver finds five consecutive incoming 1 bits, followed by a 0 bit, it destuffs the 0 bit.

Advantages:

- The boundary between two frames is unambiguously recognized by the flag pattern (flag sequences can only occur at frame boundaries and never within the data).
- Frames can contain an arbitrary number of bits made up of units of any size.
- Ensures a minimum density of transitions that help the physical layer maintain synchronization.

Both byte and bit stuffing:

- Are completely transparent to the network layer in both computers.
- Have a frame length that depends on the contents of the data.

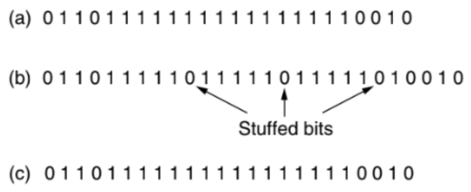

(a) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0
(b) 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 0 0 1 0
Stuffed bits
(c) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

Figure 5. Bit stuffing. (a) The original data. (b) The data as they appear on the line. (c) The data as they are stored in the receiver's memory after destuffing.

Many data link protocols use a combination of these methods for safety.

1.3 Error detection

1.4 Automatic Repeat reQuest (ARQ)

An error-control method for data transmission that uses acknowledgements (messages indicating whether or not the message has been correctly received) and timeouts to achieve reliable data transmission over an unreliable service. This mechanisms automatically request the retransmission of:

- Missing packets.
- Packets with errors.

There are three common ARQ schemes:

1.4.1 Stop and Wait

- Sender transmits information frame I and waits for positive confirmation ACK from receiver.
- Receiver receives I frame:
  - If I frame has no error sends ACK.
  - If I frame has error sends NACK.
- Sender receives I frame:
  - If ACK, proceeds and transmits new frame.
  - If NACK, retransmits frame I.
- If I, ACK or NACK is lost a timeout is required!

Issue: If the transmitter times-out and sends a packet twice, the receiver cannot tell whether the second frame is a retransmission or a new frame transmission.

Solution: Define a 1 bit sequence number in the header of the frame.

This sequence number alternates (from 0 to 1) in subsequent frames. The transmitter sends a frame with a sequence number attached to it so the receiver can check if it matches the expected. When the receiver sends an ACK, it includes the sequence number of the next packet it expects. This way, the receiver can detect duplicated frames by checking if the frame sequence numbers alternate.

Efficiency: STILL WORKING ON Inefficient when Tprop > Tf (a > 1). Sends only one frame per Round-Trip Time ($RTT = 2 * Tprop + Tf$).

1.4.2 Go Back N

Allows the transmission of new packets before earlier ones are acknowledged.

Sender:

- May transmit up to W frames without receiving RR(Receiver Ready = ACK).
- I frames are numbered sequentially I(NS): I(0), I(1), I(2), etc.
- Cannot send I(NS=i+W) until it has received the RR(NR=i).

Receiver:

- Does not accept frames out of sequence.
- Sends RR(NR) to sender indicating:
  - That all the packets up to NR-1 have been received in sequence.
  - The sequence number, NR, of the next expected frame.

Behaviour under Errors

- Frame with errors is silently discarded by the Receiver.
- If Receiver receives Data frame out of sequence:
  - First out-of-sequence-frame: Receiver sends REJ(NR) where NR = next in-sequence frame expected.
  - Following out-of sequence-frames: Receiver discards them; no REJ sent.
- When Sender receives REJ(NR=x), the Sender:
  - Goes-Back and retransmits I(x), I(x+1), etc.
  - Continues using Sliding Window mechanism.
- If timeout occurs, the Sender:
  - Requests the Receiver to send a RR message.
  - Sends a special message (RR command message).

Efficiency:

- If $W \geq 1 + 2 * a \Rightarrow S = 1$.
- If $W < 1 + 2 * a \Rightarrow S = W \div (1 + 2 * a)$.

$$S = \begin{cases} \dfrac{1 - p_e}{1 + 2ap_e} & ,W \geq 1 + 2a \\ \dfrac{W(1 - p_e)}{(1 + 2a)(1 - p_e + Wp_e)} & ,W < 1 + 2a \end{cases}$$

Figura 1: pe - frame error probability (ratio, FER)

1.4.3 Selective Repeat

Similar to **Go Back N**, however it does not discard successful frames when errors occur.

Receiver:

- Accepts out of sequence frames.
- Confirms negatively, SREJ, a frame not arrived.
- Uses RR to confirm blocks of frames arrived in sequence.

Sender: Retransmits only the frames signaled by SREJ.

STILL WORKING ON
Adequate if W (a) is very large.
Maximum window size $W = M \div 2 = 2^(k - 1)$.

$$S = \begin{cases} 1 - p_e & ,W \geq 1 + 2a \\ \dfrac{W(1 - p_e)}{1 + 2a} & ,W < 1 + 2a \end{cases}$$

Figura 2: pe - frame error probability (ratio, FER)

1.5 Framing, Error detection and ARQ in common networks

1.6 Reliability in the Protocol Stack