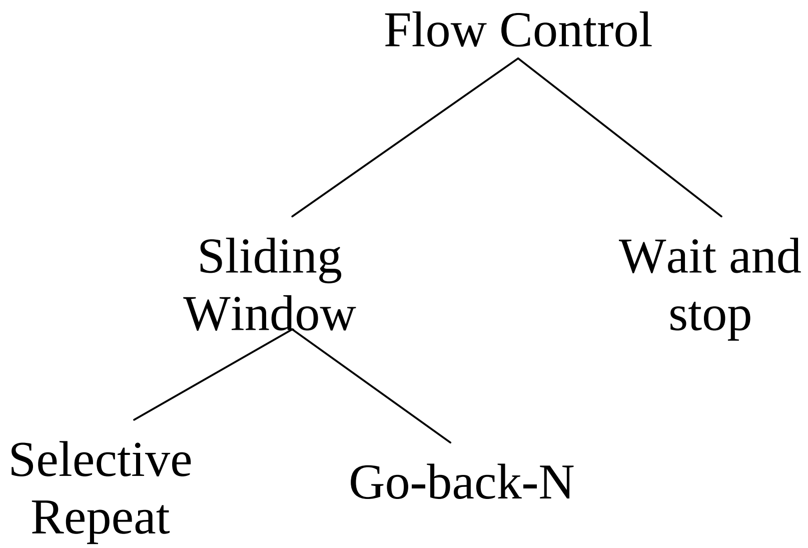

Flow Control
Sliding Window
Wait and stop
Selective Repeat
Go-back-N

DATA LINK PROTOCOLS

1- An Unrestricted Simplex Protocol

Assupmtions:

1- Data are transmitted in one direction
2- Both transmitting & receiving network lagers are always ready
3- Processing time can be ignored
4- Infinite buffer space is available
5- The communication channel between the data link layers never damages or losses frames

UTOPIA

Protocol of the sender

1- Go fetch a packet from the network layer
2- Construct an out bound frame
3- Send the frame on its way

Protocol of the receiver

1- Wait for something to happen
2- The frame arrives and it is removed from the hardware buffer
3- The data portion is passed on to the network layer
4- The data link layer settles back to wait for the next frame

2- A simplex stop-and-wait protocol

Drop the assumption of having an infinite amount of buffer space at the receiving data link layer

i) in case of synchronous transmission the sender simply inserts a delay into the previous protocol to slow it down sufficiently to keep from swamping the receiver
   This approach is too conservative
   so the solution is Asynchronous transmission

ii) Acknowledgement
   Receiver gives the sender the permission to transmit the next frame
   STOP-and-WAIT

Stop-and-go Protocol

- The simplest form of flow control
  - A sender waits after transmitting each packet
  - When the receiver is ready for another packet, the receiver sends a control message, usually a form of acknowledgement
- A half-duplex physical channel would be sufficient for this protocol
- The protocol can prevent data overrun, however, it can cause extremely inefficient use of network bandwidth

3- A simplex protocol for a noisy channel

Drop the assumption of error free channel
Frames may be either damaged or lost completely

In case of a damaged frame:

- the receiver detects this from check sum
- discard damaged frame

In case of lost frame:

- Sender has to wait for reply forever and the protocol fails
  So ADD “TIMER”

Protocol:

1- If correct frame → Acknowledgement → next
2- If damaged frame → discard it (receiver) → time out (sender)
3- If correct frame but lost Acknowledgement → time out → send frame again → Duplicate frame

Thus the protocol fails.
Solution:
Put a “sequence number” in the header of each frame it sends. Then the receiver checks the sequence number to see if it is new or duplicate to be discarded

Protocols with Acknowledgement + time out + sequence number are sometimes called:
PAR (Positive Acknowledgement with Retransmission)
Or ARQ (Automatic Repeat request)

Note: time out interval is required to be long enough to prevent premature time outs

4- Sliding Window Protocol

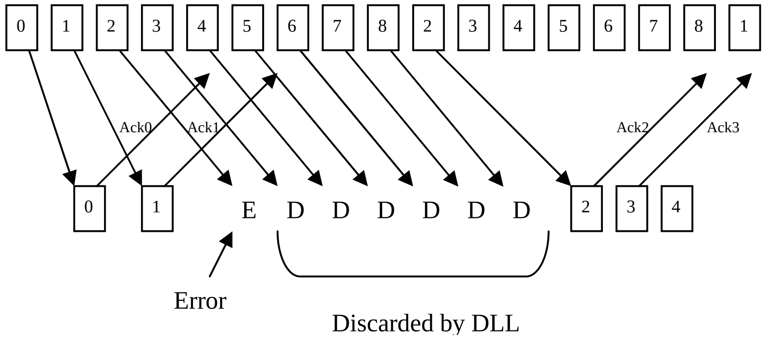
Sliding window mechanism is widely used to integrate error control and flow control in a convenient way

- In most practical situations, there is a need for transmitting data in both directions (Full-duplex transmission)
- Piggy backing technique
- The essence of all sliding window protocols:
    - The sender maintains a set of sequence numbers corresponding to frames it is permitted to send
    - The receiver maintains a “receiving window” corresponding to the set of frames it is permitted to accept
- The receiving date link layer’s window corresponds to the frames it may accept
- Any frame falling outside the window is discarded without comment
- When a frame whose sequence number is equal to the lower edge of the window is received, it is passed to the network layer, an acknowledgement is generated, and the window is rotated by one
- Unlike the sender’s window, the receiver’s window always remains at its initial size
- The sequence number within the sender’s window represents frames sent but is yet not acknowledged
- The upper edge of the window is advanced by one whenever a new packet arrives from the network layer, and it is given the next highest sequence number
- The lower edge is advanced by one when an acknowledgement comes in
- The sender needs n buffers if the maximum window size is n
- If the window ever grows to its maximum size, the sending data link layer must forcibly shut off the network layer until another buffer comes free

Go Back n Protocol

It works well if errors are rare.

The receiver discards all subsequent frames, sending no acknowledgements for the discarded frames



Selective repeat protocol

The receiving DLL has to store all the correct frames following the bad one. When the sender finally notices that something is wrong, it just retransmits the one bad frame not all its successors

It requires DLL memory