**Flow control (data)**

From Wikipedia, the free encyclopedia

[Jump to navigation](https://en.wikipedia.org/wiki/Flow_control_(data)#mw-head) [Jump to search](https://en.wikipedia.org/wiki/Flow_control_(data)#searchInput)

Not to be confused with [Control flow](https://en.wikipedia.org/wiki/Control_flow).

In [data communications](https://en.wikipedia.org/wiki/Data_communications), **flow control** is the process of managing the rate of data transmission between two nodes to prevent a fast sender from overwhelming a slow receiver. It provides a mechanism for the receiver to control the transmission speed, so that the receiving node is not overwhelmed with data from transmitting node. Flow control should be distinguished from [congestion control](https://en.wikipedia.org/wiki/Congestion_control), which is used for controlling the flow of data when congestion has actually occurred.[[1]](https://en.wikipedia.org/wiki/Flow_control_(data)#cite_note-ATM-Traffic-Management-1) Flow control mechanisms can be classified by whether or not the receiving node sends feedback to the sending node.

Flow control is important because it is possible for a sending computer to transmit information at a faster rate than the destination computer can receive and process it. This can happen if the receiving computers have a heavy traffic load in comparison to the sending computer, or if the receiving computer has less processing power than the sending computer.



**Contents**

* [1 Stop-and-wait](https://en.wikipedia.org/wiki/Flow_control_(data)#Stop-and-wait)
  + [1.1 Operations](https://en.wikipedia.org/wiki/Flow_control_(data)#Operations)
  + [1.2 Pros and cons of stop and wait](https://en.wikipedia.org/wiki/Flow_control_(data)#Pros_and_cons_of_stop_and_wait)
* [2 Sliding Window](https://en.wikipedia.org/wiki/Flow_control_(data)#Sliding_Window)
  + [2.1 Go Back N](https://en.wikipedia.org/wiki/Flow_control_(data)#Go_Back_N)
  + [2.2 Selective Repeat](https://en.wikipedia.org/wiki/Flow_control_(data)#Selective_Repeat)
* [3 Comparison](https://en.wikipedia.org/wiki/Flow_control_(data)#Comparison)
  + [3.1 Stop-and-Wait](https://en.wikipedia.org/wiki/Flow_control_(data)#Stop-and-Wait_2)
  + [3.2 Selective Repeat](https://en.wikipedia.org/wiki/Flow_control_(data)#Selective_Repeat_2)
* [4 Transmit flow control](https://en.wikipedia.org/wiki/Flow_control_(data)#Transmit_flow_control)
  + [4.1 Hardware flow control](https://en.wikipedia.org/wiki/Flow_control_(data)#Hardware_flow_control)
  + [4.2 Software flow control](https://en.wikipedia.org/wiki/Flow_control_(data)#Software_flow_control)
* [5 Open-loop flow control](https://en.wikipedia.org/wiki/Flow_control_(data)#Open-loop_flow_control)
* [6 Closed-loop flow control](https://en.wikipedia.org/wiki/Flow_control_(data)#Closed-loop_flow_control)
* [7 See also](https://en.wikipedia.org/wiki/Flow_control_(data)#See_also)
* [8 References](https://en.wikipedia.org/wiki/Flow_control_(data)#References)
* [9 External links](https://en.wikipedia.org/wiki/Flow_control_(data)#External_links)

**Stop-and-wait**

Main article: [Stop-and-wait ARQ](https://en.wikipedia.org/wiki/Stop-and-wait_ARQ)

Stop-and-wait flow control is the simplest form of flow control. In this method the message is broken into multiple frames, and the receiver indicates its readiness to receive a frame of data. The sender waits for a receipt acknowledgement (ACK) after every frame for a specified time (called a time out). The receiver sends the ACK to let the sender know that the frame of data was received correctly. The sender will then send the next frame only after the ACK.

**Operations**

1. **Sender:** Transmits a single frame at a time.
2. Sender waits to receive ACK within time out.
3. **Receiver:** Transmits acknowledgement (ACK) as it receives a frame.
4. Go to step 1 when ACK is received, or time out is hit.

If a frame or ACK is lost during transmission then the frame is re-transmitted. This re-transmission process is known as ARQ (automatic repeat request).

The problem with Stop-and-wait is that only one frame can be transmitted at a time, and that often leads to inefficient transmission, because until the sender receives the ACK it cannot transmit any new packet. During this time both the sender and the channel are unutilised.

**Pros and cons of stop and wait**

**Pros**

The only advantage of this method of flow control is its simplicity.

**Cons**

The sender needs to wait for the ACK after every frame it transmits. This is a source of inefficiency, and is particularly bad when the [propagation delay](https://en.wikipedia.org/wiki/Propagation_delay) is much longer than the [transmission delay](https://en.wikipedia.org/wiki/Transmission_delay).[[2]](https://en.wikipedia.org/wiki/Flow_control_(data)#cite_note-PC-Radio_Flow_Control-2)

Stop and wait can also create inefficiencies when sending longer transmissions.[[3]](https://en.wikipedia.org/wiki/Flow_control_(data)#cite_note-ak2-3) When longer transmissions are sent there is more likely chance for error in this protocol. If the messages are short the errors are more likely to be detected early. More inefficiency is created when single messages are broken into separate frames because it makes the transmission longer.[[4]](https://en.wikipedia.org/wiki/Flow_control_(data)#cite_note-lwilliam-4)

**Sliding Window**

Main article: [Sliding Window Protocol](https://en.wikipedia.org/wiki/Sliding_Window_Protocol)

A method of flow control in which a receiver gives a transmitter permission to transmit data until a window is full. When the window is full, the transmitter must stop transmitting until the receiver advertises a larger window.[[5]](https://en.wikipedia.org/wiki/Flow_control_(data)#cite_note-Sliding_Window-5)

Sliding-window flow control is best utilized when the buffer size is limited and pre-established. During a typical communication between a sender and a receiver the receiver allocates buffer space for *n* frames (*n* is the buffer size in frames). The sender can send and the receiver can accept *n* frames without having to wait for an acknowledgement. A sequence number is assigned to frames in order to help keep track of those frames which did receive an acknowledgement. The receiver acknowledges a frame by sending an acknowledgement that includes the sequence number of the next frame expected. This acknowledgement announces that the receiver is ready to receive n frames, beginning with the number specified. Both the sender and receiver maintain what is called a window. The size of the window is less than or equal to the buffer size.

Sliding window flow control has far better performance than stop-and-wait flow control. For example, in a wireless environment if data rates are low and noise level is very high, waiting for an acknowledgement for every packet that is transferred is not very feasible. Therefore, transferring data as a bulk would yield a better performance in terms of higher throughput.

Sliding window flow control is a point to point protocol assuming that no other entity tries to communicate until the current data transfer is complete. The window maintained by the sender indicates which frames it can send. The sender sends all the frames in the window and waits for an acknowledgement (as opposed to acknowledging after every frame). The sender then shifts the window to the corresponding sequence number, thus indicating that frames within the window starting from the current sequence number can be sent.

**Go Back N**

Main article: [Go-Back-N ARQ](https://en.wikipedia.org/wiki/Go-Back-N_ARQ)

An automatic repeat request (ARQ) algorithm, used for error correction, in which a negative acknowledgement (NAK) causes retransmission of the word in error as well as the next N–1 words. The value of N is usually chosen such that the time taken to transmit the N words is less than the round trip delay from transmitter to receiver and back again. Therefore, a buffer is not needed at the receiver.

The normalized propagation delay (a) = ​propagation time (Tp)⁄transmission time (Tt), where Tp = Length (L) over propagation velocity (V) and Tt = bitrate (r) over Framerate (F). So that a =​LF⁄Vr.

To get the utilization you must define a window size (N). If N is greater than or equal to 2a + 1 then the utilization is 1 (full utilization) for the transmission channel. If it is less than 2a + 1 then the equation ​N⁄1+2a must be used to compute utilization.[[6]](https://en.wikipedia.org/wiki/Flow_control_(data)#cite_note-Go_Back_N-6)

**Selective Repeat**

Main article: [Selective Repeat ARQ](https://en.wikipedia.org/wiki/Selective_Repeat_ARQ)

Selective Repeat is a connection oriented protocol in which both transmitter and receiver have a window of sequence numbers. The protocol has a maximum number of messages that can be sent without acknowledgement. If this window becomes full, the protocol is blocked until an acknowledgement is received for the earliest outstanding message. At this point the transmitter is clear to send more messages.[[7]](https://en.wikipedia.org/wiki/Flow_control_(data)#cite_note-Selective_Repeat-7)

**Comparison**

This section is geared towards the idea of comparing [Stop-and-wait](https://en.wikipedia.org/wiki/Stop-and-wait_ARQ), [Sliding Window](https://en.wikipedia.org/wiki/Sliding_window) with the subsets of [Go Back N](https://en.wikipedia.org/wiki/Go-Back-N_ARQ) and [Selective Repeat](https://en.wikipedia.org/wiki/Selective_Repeat_ARQ).

**Stop-and-Wait**

Error free: 1 2 a + 1 {\displaystyle {\frac {1}{2a+1}}} .[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

With errors: 1 − P 2 a + 1 {\displaystyle {\frac {1-P}{2a+1}}} .[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

**Selective Repeat**

We define throughput T as the average number of blocks communicated per transmitted block. It is more convenient to calculate the average number of transmissions necessary to communicate a block, a quantity we denote by 0, and then to determine T from the equation T = 1 b {\displaystyle T={\frac {1}{b}}} .[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

**Transmit flow control**

Transmit flow control may occur:

* between [data terminal equipment](https://en.wikipedia.org/wiki/Data_terminal_equipment) (DTE) and a [switching center](https://en.wikipedia.org/wiki/Switching_center), via [data circuit-terminating equipment](https://en.wikipedia.org/wiki/Data_circuit-terminating_equipment) (DCE), the opposite types interconnected straightforwardly,
* or between two devices of the same type (two DTEs, or two DCEs), interconnected by a [crossover cable](https://en.wikipedia.org/wiki/Crossover_cable).

The [transmission rate](https://en.wikipedia.org/wiki/Bit_rate) may be controlled because of [network](https://en.wikipedia.org/wiki/Telecommunications_network) or DTE requirements. Transmit flow control can occur independently in the two directions of data transfer, thus permitting the transfer rates in one direction to be different from the transfer rates in the other direction. Transmit flow control can be

* either [stop-and-wait](https://en.wikipedia.org/wiki/Stop-and-wait_ARQ),
* or use a [sliding window](https://en.wikipedia.org/wiki/Sliding_window).

Flow control can be performed

* either by [control signal](https://en.wikipedia.org/wiki/Control_signal) lines in a data communication interface (see [serial port](https://en.wikipedia.org/wiki/Serial_port) and [RS-232](https://en.wikipedia.org/wiki/RS-232)),
* or by reserving in-band control characters to signal flow start and stop (such as the [ASCII](https://en.wikipedia.org/wiki/ASCII) codes for [XON/XOFF](https://en.wikipedia.org/wiki/XON/XOFF)).

**Hardware flow control**

In common RS-232 there are pairs of control lines which are usually referred to as *hardware flow control*:

* RTS (Request To Send) and CTS (Clear To Send), used in [RTS flow control](https://en.wikipedia.org/wiki/RS-232_RTS/CTS)
* DTR ([Data Terminal Ready](https://en.wikipedia.org/wiki/Data_Terminal_Ready)) and DSR ([Data Set Ready](https://en.wikipedia.org/wiki/Data_Set_Ready)), DTR flow control

Hardware flow control is typically handled by the DTE or "master end", as it is first raising or asserting its line to command the other side:

* In the case of RTS control flow, DTE sets its RTS, which signals the opposite end (the slave end such as a DCE) to begin monitoring its data input line. When ready for data, the slave end will raise its complementary line, CTS in this example, which signals the master to start sending data, and for the master to begin monitoring the slave's data output line. If either end needs to stop the data, it lowers its respective "data readiness" line.
* For PC-to-modem and similar links, in the case of DTR flow control, DTR/DSR are raised for the entire modem session (say a dialup internet call where DTR is raised to signal the modem to dial, and DSR is raised by the modem when the connection is complete), and RTS/CTS are raised for each block of data.

An example of hardware flow control is a [Half-duplex](https://en.wikipedia.org/wiki/Half-duplex) radio modem to computer interface. In this case, the controlling software in the modem and computer may be written to give priority to incoming radio signals such that outgoing data from the computer is paused by lowering CTS if the modem detects a reception.

* Polarity:
  + RS-232 level signals are inverted by the driver ICs, so line polarity is TxD-, RxD-, CTS+, RTS+ (Clear to send when HI, Data 1 is a LO)
  + for microprocessor pins the signals are TxD+, RxD+, CTS-, RTS- (Clear to send when LO, Data 1 is a HI)

**Software flow control**

Main article: [Software flow control](https://en.wikipedia.org/wiki/Software_flow_control)

Conversely, XON/XOFF is usually referred to as software flow control.

**Open-loop flow control**

The open-loop flow control mechanism is characterized by having no feedback between the receiver and the transmitter. This simple means of control is widely used. The allocation of resources must be a "prior reservation" or "hop-to-hop" type.

Open-loop flow control has inherent problems with maximizing the utilization of network resources. Resource allocation is made at connection setup using a CAC (Connection Admission Control) and this allocation is made using information that is already "old news" during the lifetime of the connection. Often there is an over-allocation of resources and reserved but unused capacities are wasted. Open-loop flow control is used by [ATM](https://en.wikipedia.org/wiki/Asynchronous_Transfer_Mode) in its [CBR](https://en.wikipedia.org/wiki/Constant_Bit_Rate), [VBR](https://en.wikipedia.org/wiki/Variable_bitrate) and [UBR](https://en.wikipedia.org/wiki/Unspecified_Bit_Rate) services (see [traffic contract](https://en.wikipedia.org/wiki/Traffic_contract) and [congestion control](https://en.wikipedia.org/wiki/Congestion_control)).[[1]](https://en.wikipedia.org/wiki/Flow_control_(data)#cite_note-ATM-Traffic-Management-1)

Open-loop flow control incorporates two controls; the controller and a regulator. The regulator is able to alter the input variable in response to the signal from the controller. An open-loop system has no feedback or feed forward mechanism, so the input and output signals are not directly related and there is increased traffic variability. There is also a lower arrival rate in such system and a higher loss rate. In an open control system, the controllers can operate the regulators at regular intervals, but there is no assurance that the output variable can be maintained at the desired level. While it may be cheaper to use this model, the open-loop model can be unstable.

**Closed-loop flow control**

The closed-loop flow control mechanism is characterized by the ability of the network to report pending [network congestion](https://en.wikipedia.org/wiki/Network_congestion) back to the transmitter. This information is then used by the transmitter in various ways to adapt its activity to existing network conditions. Closed-loop flow control is used by [ABR](https://en.wikipedia.org/wiki/Available_Bit_Rate) (see [traffic contract](https://en.wikipedia.org/wiki/Traffic_contract) and [congestion control](https://en.wikipedia.org/wiki/Congestion_control)).[[1]](https://en.wikipedia.org/wiki/Flow_control_(data)#cite_note-ATM-Traffic-Management-1) Transmit flow control described above is a form of closed-loop flow control.

This system incorporates all the basic control elements, such as, the sensor, transmitter, controller and the regulator. The sensor is used to capture a process variable. The process variable is sent to a transmitter which translates the variable to the controller. The controller examines the information with respect to a desired value and initiates a correction action if required. The controller then communicates to the regulator what action is needed to ensure that the output variable value is matching the desired value. Therefore, there is a high degree of assurance that the output variable can be maintained at the desired level. The closed-loop control system can be a feedback or a feed forward system:

A feedback closed-loop system has a feed-back mechanism that directly relates the input and output signals. The feed-back mechanism monitors the output variable and determines if additional correction is required. The output variable value that is fed backward is used to initiate that corrective action on a regulator. Most control loops in the industry are of the feedback type.

In a feed-forward closed loop system, the measured process variable is an input variable. The measured signal is then used in the same fashion as in a feedback system.

The closed-loop model produces lower loss rate and queuing delays, as well as it results in congestion-responsive traffic. The closed-loop model is always stable, as the number of active lows is bounded.

**See also**

* [Software flow control](https://en.wikipedia.org/wiki/Software_flow_control)
* [Computer networking](https://en.wikipedia.org/wiki/Computer_networking)
* [Traffic contract](https://en.wikipedia.org/wiki/Traffic_contract)
* [Congestion control](https://en.wikipedia.org/wiki/Congestion_control)
* [Teletraffic engineering in broadband networks](https://en.wikipedia.org/wiki/Teletraffic_engineering_in_broadband_networks)
* [Teletraffic engineering](https://en.wikipedia.org/wiki/Teletraffic_engineering)
* [Ethernet flow control](https://en.wikipedia.org/wiki/Ethernet_flow_control)
* [Handshaking](https://en.wikipedia.org/wiki/Handshaking)

**References**

 [Network Testing Solutions, ATM Traffic Management White paper](https://web.archive.org/web/20020910153543/http:/www.parallaxresearch.com/dataclips/pub/infotech/protocols/ATM/ATM_Traffic_Mgmt.PDF) last accessed 15 March 2005.

  [*"ERROR CONTROL"*](http://www.theparticle.com/cs/bc/net/flowctrl.pdf) *(PDF). 28 September 2005. Retrieved 10 November 2018.*

  *arun (20 November 2012).* [*"Flow Control Techniques"*](http://www.angelfire.com/ak2/wireless/flowctrl.html)*. angelfire.com. Retrieved 10 November 2018.*

  [*"last accessed 1 December 2012"*](http://people.bridgewater.edu/~lwilliam/Chapter%2005/sld053.htm)*. people.bridgewater.edu. 1 December 2012. Retrieved 10 November 2018.*

  [Webster Dictionary definition](http://www.webster-dictionary.org/definition/sliding-window) last accessed 3 December 2012.

  [Focal Dictionary of Telecommunications, Focal Press](http://www.credoreference.com/entry/bhfidt/go_back_n_arq) last accessed 3 December 2012.

* 1.  [Data Transmission over Adpative HF Radio Communication Systems using Selective Repeat Protocol](https://docs.google.com/viewer?a=v&q=cache:rp6GPovdFiIJ:facta.junis.ni.ac.rs/eae/fu2k02/fu06.pdf+&hl=en&gl=us&pid=bl&srcid=ADGEESiFJNDv8E5p9LS_AHD-zZHMUUrWx5FRTU7xolQL_D58JT3mHXPwldc0CQu32LFShXmnc0MleeH6GUvw0qL3jGuxlwlh_SYoA2h0NZqOruQA3mUXAcEK7YW7_lbx_FSKP1ou5473&sig=AHIEtbTxR21FrNmWYX-vN8cV0S3hi4yg2A) last accessed 3 December 2012.

Sliding window:

* [[1]](http://www.cncroutersource.com/closed-loop-system.html) last accessed 27 November 2012.

**External links**

* [RS-232 flow control and handshaking](https://www.lammertbies.nl/comm/info/RS-232_flow_control.html)

In a network, the sender sends the data and the receiver receives the data. But suppose a situation where the sender is sending the data at a speed higher than the receiver is able to receive and process it, then the data will get lost. **Flow-control** methods will help in ensuring this. The flow control method will keep a check that the senders send the data only at a speed that the receiver is able to receive and process. So, let's get started with the blog and learn more about flow control.

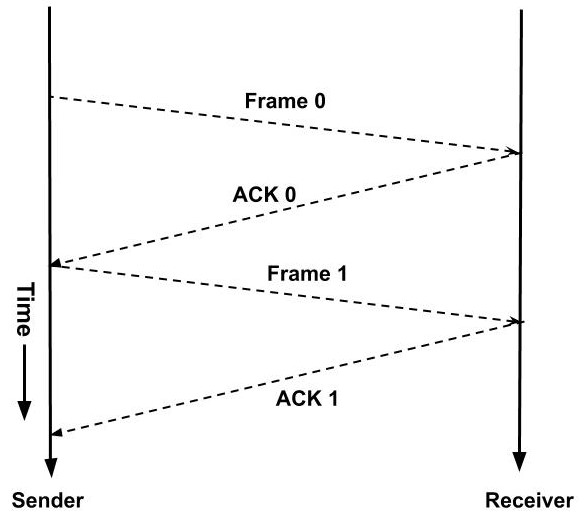
#### Flow Control

Flow control tells the sender how much data should be sent to the receiver so that it is not lost. This mechanism makes the sender wait for an acknowledgment before sending the next data. There are two ways to control the flow of data:

1. Stop and Wait Protocol
2. Sliding Window Protocol

#### Stop and Wait Protocol

It is the simplest flow control method. In this, the sender will send one frame at a time to the receiver. Until then, the sender will **stop and wait** for the acknowledgment from the receiver. When the sender gets the acknowledgment then it will send the next data packet to the receiver and wait for the acknowledgment again and this process will continue. This can be understood by the diagram below.



Suppose if any frame sent is not received by the receiver and is lost. So the receiver will not send any acknowledgment as it has not received any frame. Also, the sender will not send the next frame as it will wait for the acknowledgment for the previous frame which it had sent. So a deadlock situation can be created here. To avoid any such situation there is a time-out timer. The sender will wait for this fixed amount of time for the acknowledgment and if the acknowledgment is not received then it will send the frame again.

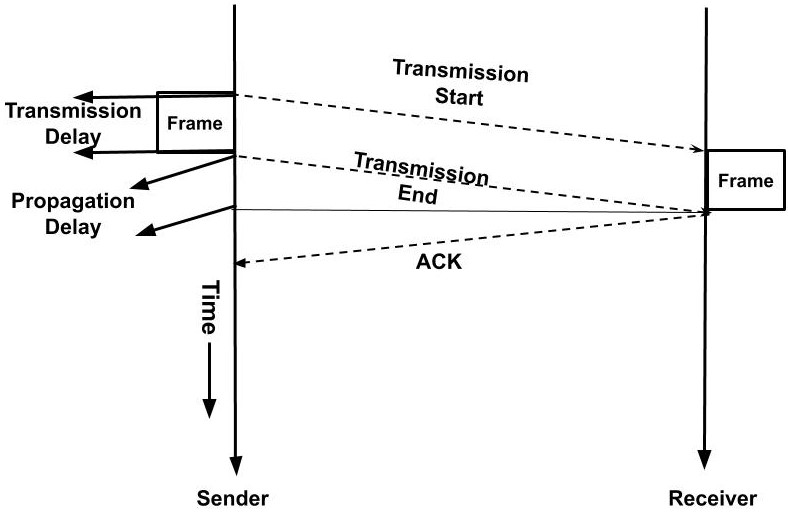
There are two types of delays while sending these frames:

* **Transmission Delay:** Time taken by the sender to send all the bits of the frame onto the wire is called transmission delay. This is calculated by dividing the data size(D) which has to be sent by the bandwidth(B) of the link.

Td = D / B

* **Propagation Delay:** Time taken by the last bit of the frame to reach from one side to the other side is called propagation delay. It is calculated by dividing the distance between the sender and receiver by the wave propagation speed.

Tp = d / s ; where d = distance between sender and receiver, s = wave propagation speed



The propagation delay for sending the data frame and the acknowledgment frame is the same as distance and speed will remain the same for both frames. Hence, the total time required to send a frame is

Total time= Td(Transmission Delay) + Tp(Propagation Delay for data frame) + Tp(Propagation Delay for acknowledgment frame)

The sender is doing work only for **Td** time and for the rest **2Tp** time the sender is waiting for the acknowledgment.

Efficiency = Useful Time/ Total Time

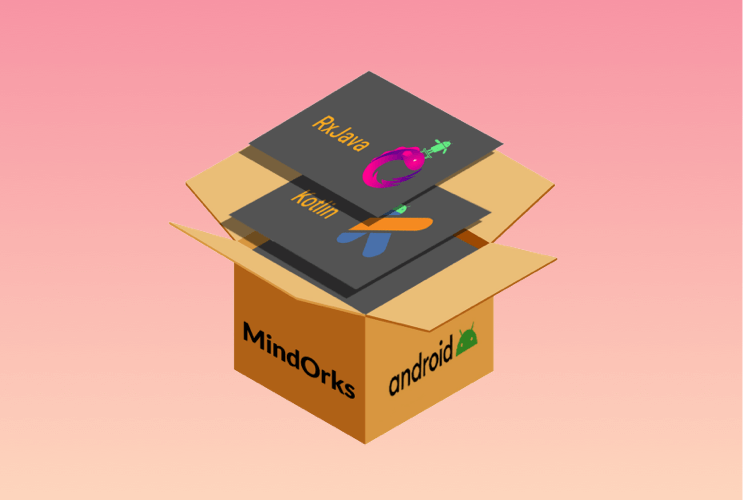
**η=Td / (Td+2Tp)**

#### Advantages of Stop and Wait Protocol

1. It is very simple to implement.

#### Disadvantages of Stop and Wait Protocol

1. We can send only one packet at a time.
2. If the distance between the sender and the receiver is large then the propagation delay would be more than the transmission delay. Hence, efficiency would become very low.
3. After every transmission, the sender has to wait for the acknowledgment and this time will increase the total transmission time.



NEW

##### Courses All-Access Subscription

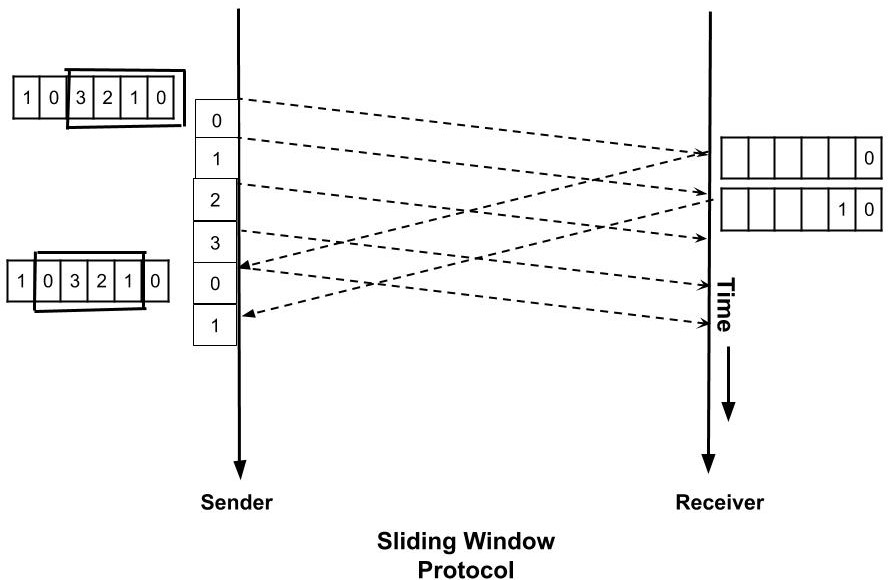
Learn all the in demand coding skills you need to excel in your career and get mentorship for courses. Now just ~~₹10980~~ ₹5490 Monthly

#### Sliding Window Protocol

As we saw that the disadvantage of the stop and wait protocol is that the sender waits for the acknowledgment and during that time the sender is idle. In sliding window protocol we will utilize this time. We will change this waiting time into transmission time.

A **window** is a buffer where we store the frames. Each frame in a window is numbered. If the window size is **n** then the frames are numbered from the number 0 to n-1. A sender can send n frames at a time. When the receiver sends the acknowledgment of the frame then we need not store that frame in our window as it has already been received by the receiver. So, the window in the sender side **slides** to the next frame and this window will now contain a new frame along with all the previous unacknowledged frames of the window. **At any instance of time window will only contain the unacknowledged** **frames.** This can be understood with the ***example*** below:

1. Suppose the size of the window is 4. So, the frames would be numbered as 0,1,2,3,0,1,2,3,0,… so on.
2. Initially, the frames in the window are 0,1,2, 3. Now, the sender starts transmitting the frames. The first frame is sent, then second and so on.
3. When the receiver receives the first frame i.e. frame 0. Then it sends an acknowledgment.
4. When the acknowledgment is received by the sender then it knows that the first frame has been received by the receiver and it need not keep its record. So, the **window slides** to the next frame.
5. The new window contains the frame 1, 2, 3, 0. In this way, the window slides hence the name sliding window protocol.



Using sliding window protocol, the efficiency can be made maximum i.e. 1. In sliding window protocol we are using the propagation delay time also for the transmission. For doing this we the sender should be sending the data frame all the time i.e for Td+2Tp time. So, **what should be the number of packets such that the efficiency is maximum?**

We will apply a simple unitary method to find this. In Td units of time, we can send one packet. So in one unit of time, we can send 1/Td packets. We have total time as Td+2Tp. Therefore, in Td+2Tp time we can send (Td+2Tp)/Td packets. Let a=Tp/Td. So, if we send 1+2a packets then the efficiency is 1.

Td units of time → 1 packet transmitted

1 unit of time → (1/ Td) packet transmitted

Td + 2Tp units of time → (Td + 2Tp) / Td packets transmitted

This is how the flow of data is controlled using the above two mechanisms. Hope you learned something new today.

Do share this blog with your friends to spread the knowledge. Visit our [YouTube channel](https://www.youtube.com/afteracademy) for more content. You can read more blogs from [here](https://afteracademy.com/blogs).

Keep Learning :)