

Implementation of Data Link Control Protocols in Wired Network

Sudhanshu Maurya^{#1}, Vikas Kumar Nayak^{*2}, Dr. A Nagaraju^{#3}

[#]Assistant Professor, Department of Computer Science & Engineering, Jharkhand Rai University
Ranchi, Jharkhand, India

Abstract— Data Link Layer is the protocol layer which transfers data between connected networks. It provides the functional and procedural means to transfer data between network and its nodes. It also provide the means to rectify the errors and usually correct it that may occur in the Physical Layer. Study and simulate protocol to analyse and find the advantages and disadvantages to improve the quality of protocol such that it handles the network data transmission in proper way. The protocol is studied so that the details of the protocols are revealed and the limitations of the protocol can be overcome later. In this research paper we give a brief introduction about Data Link Layer Protocols and introduced a new protocol (Protocol 7) to overcome the problem of previous protocols on the basis of priority of the frame.

Keywords— Protocol, Adjacent network, Simulate, Transmission, Segment, Frame.

I. INTRODUCTION

When we communicate with someone, we are sharing our information. It is exchange of data between two devices via some form of transmission medium such as a wired cable. The data communication system consists of five components i.e. Sender, Receiver, Message, Transmission medium and Protocol. Sender sends the message to other nodes of the network. Receiver receives the message from sender. Message is data that is to be communicated. Transmission medium is a path by which message travels from one node to another. Protocol is a set of rules that helps to control data communication between communication devices [1]. OSI (Open System Interconnection) model is commonly used to specify and describe computer communication protocol [2]. It was first taken into consideration in late 1970s. The motive of the OSI model is to show how to facilitate communication between different systems. It is not a protocol; it is a model for designing and understanding network architecture. The second layer of OSI model is Data Link Layer it handles a system which have two nodes connected via a physical link. The major work of this layer is to establish a connection between the nodes of network, one for sending and other for receiving. Data Link Layer has two sub layers: Data Link Control and Media Access Control. The data link layer is responsible for designing and communication between two contiguous nodes.

II. PROTOCOLS OF DATA LINK CONTROL PROTOCOL

Data Link Control Protocol consists of six protocols, two for noiseless channel i.e. ideal channel and four for noisy

channel i.e. real channel. The first category of protocols is not actually implemented because ideal channel does not exist in present scenario. The list of protocols is: Unrestricted Simplex Protocol, Stop and Wait, Positive Acknowledgment with Retransmission, Sliding Window Protocols, Go-Back-N, and Selective Repeat [10].

A. Unrestricted Simplex Protocol

It is a simple protocol with no restrictions. It has no error control and flow control. This protocol provides data transmission in only one direction only i.e from sender to receiver. We assume that the communication channel is to be error free and the receiver is able to process all the input infinitely quickly. The sender pumps the data as fast as it can. The problem with this protocol is it assumes an error free communication channel [3]. So there is no provision of detection and correction of errors in data frame.

B. Stop and Wait

In Stop and Wait protocol the sender takes packets from network layer. Copies the packet into frames and transmit them. After this the sender busy waits for an ACK from receiver side. The receiver is busy and waits until a frame is not received from sender side. When frame is received it passes the data packet to network layer and sends an ACK to the sender for the frame is successfully received. Then it back to waiting stage and this process is continues till end of file. Here only one outstanding frame at a time so no sequence number is required. In this protocol it is easy to see the frame, ACK lost / damage or a deadlock situation. The main problem with this protocol is if the ACK not received by the sender then it wait for infinite time[16].

C. Possative Acknowledgement with Retransmission

Possative Acknowledgement with Retransmission (PAR) protocol is updated version of Stop and Wait Protocol. It is a Stop and Wait Protocol with ARQ (Automatic Repeat Request). ARQ protocols are mainly used in communication system for their easy implementation and good reliability [5][13]. In this protocol there is only one outstanding frame at a time so sequence number used to determine any lost or damaged frame and sequence number does not respond until a positive acknowledgement is not received. Here only '0' and '1' is used as a sequence number. It can't change to other sequence number until the correct one is received. The ACK frame sent back is marked as an ACK frame as it contains the

sequence number that is being acknowledged. It also keeps the sender and receiver synchronized. PAR also handles the congestion, lost/damaged frame as data frames will be resent and this process is going on until a positive acknowledgement is not received. The main problem with this protocol is when frames and ACK are lost, the number of frames sent to transfer the entire message become very large.

D. Sliding Window Protocol

In the previous protocols like stop and wait & PAR, data frames were transmitted in only one direction. In most sensible things, there's a necessity for transmittal data in both directions i.e. from sender to receiver and receiver to sender. Sliding Window protocol is popular method for both error and flow control. Many Authors [6,7,8] classify them into three parts namely, One-Bit Sliding Window Protocol, Go-Back-N, Selective Repeat with respect of their sender and receiver width. One-Bit Sliding Window protocol has the width of sender and receiver is equal to 1, Go-Back-N has width of sender window is not fixed but receiver window has 1, and the Selective Repeat protocol have the width of both sender and receiver is greater than 1 [9]. Generally any one of the two data link layers goes and transmits the first frame. The first packet from its network layer is fetched by the starting machine, a frame is built from it, and then it is sent. On arrival of this (or other) frame, it is checked by the receiving data link layer if at all it is a duplicate, just the same as in the previous protocol PAR. If it happens to be the expected frame, it is passed on to the network layer and the receiver's window slides up. The number of the last frame received without error is contained in the acknowledgement field. If this number matches with the sequence number of the frame that the sender has been trying to send, the sender comes to know about its success with the help of the frame stored in buffer and the sender can fetch the next packet from its network layer. If the sequence number fails to agree, it must keep on trying to send the same frame. On receiving a frame, the frame is also sent back [8].

E. Go-back-N

The Go Back N protocol is the advance version of PAR. By using buffers it allows the sender to have more than one outstanding frame at one time. This protocol helps the sender in maintaining a buffer of a pre-determined size. If there is any room available in the buffer it gets packed and stores it in the correct empty slot ($\text{seq_nr} \% \text{WINDOWSIZE}$). It creates a frame with the correct seq_nr and transmits it and logical timer then resets to 0. The Upper Bound of the window is then filled up by circularly incrementing the 'next_frame_to_send'. If no buffers are empty, then the physical layer is checked to see if there is any ACK present. If a good frame is received and the ACK number is within the current window then the number of buffers used is decremented and reset the logical timer (to a negative value) to show an unused slot and slide the Lower Bound of the window by circularly incrementing the ack number that is expected. This procedure runs in a loop until ack_expected equals ack_received i.e. this process helps

us in determining how many ACK have been received and how many of them have yet not been acknowledged.

If bad frames or out of window frames are detected, the logical timers then gets updated. When a frame is timed out, it gets retransmitted and is reset to 0. No ACK will arrive if a frame is timed out because of loss. So on the next iteration of the loop the next frame will be timed out and be re-sent. Hence with this, all the timed out frames and the subsequent frames will be retransmitted and with this we come to the actual definition of Go Back N. The receiver waits for the arrival of the frame. On arrival of a bad frame, it will again wait. When a good frame arrives it checks the sequence number of the arrived frame, if the expected sequence number does not arrive, it resends an ACK for the last correct sequence number that has been received successfully. If expected sequence number is obtained then it passes the packet to the Network layer and it updates the last correct sequence number received, variable and circularly increments of the next sequence number which is expected variable. An ack is then created and transmitted for the same and then it loops back to the physical layer to retrieve the next incoming frame. The Upper Bound of the window is represented by the $\text{next_frame_to_send}$ and the Lower Bound by the ack_expected . The logical timer consists of an integer type array. When the value is negative then the corresponding slot in the buffer remains unused. At any time after checking the physical layer a loop is run that increments the timers of all the used slots by 1 and It simultaneously checks if any of the timer values has reached the threshold limit i.e. timeout time. The value stored in the logical timer corresponding to a buffer slot is the time the buffered packet has iterated in the while loop.

F. Selective Repeat

The Selective Repeat protocol is the improved version of Go Back N protocol as it has buffers on both sides of the receiver as well as the sender. This protocol allows the sender to have more than one outstanding frame at a particular time and the receiver to accept out of order frames and store them in its own window [13].

Sender for this improved version is a little modified from that of Go Back N .The Maintenance of buffers and logical timers is exactly the same as in Go Back N. The only difference is that if a negative ACK is received the sender retransmits the related frame identified by the NAK. All others like timeouts, loop iterations and retransmissions are all the same as Go Back N. This is different from Go Back N in the sense that it only retransmits the frame for which a NAK is received and not all the consecutive frames, as the receiver keeps a window of frames not the whole series can be retransmitted only the timed out frames or discard frames needs to be transmitted. The receiver waits until a frame is arrived. If a bad frame arrives or timeout occurs or if an out of sequence frame arrives and if NAK has not been sent yet then a NAK is sent for the expected sequence number. If there is any room in the receiver's buffer then a packet is stored in the correct slot i.e. $\text{sequence_number} \% \text{WINDOWSIZE}$ and the slot is flagged as

used [15]. In this respect all buffered packets are passed to the network layer in sequence. If the flag that indicates whether an ACK has to be sent or not i.e. send_ack is set then an ACK is sent for the last correct sequence of frame received. Then we go back to the main loop which is waiting and start over it again.

The Selective Repeat protocol has some problem to implement. The sequence numbers is often to be bigger than the window size, ensuring there's no overlap will occur within the window [12]. This permits receiver and sender to be unbroken in synchronization even once frames and ACK are lost at awfully high rate. The buffering and ACK enable this protocol to simply handle unhealthy frames, congestion and lost frames. It had been found that a far higher timeout price is required than in return N so as to cut back the quantity of frames sent. A lower timeout price leads to too several frames temporal arrangement out and being retransmitted unnecessarily [11].

Generally any one of the two data link layers goes and transmits the first frame. The first packet from its network layer is fetched by the starting machine, a frame is built from it, and then it is sent. On arrival of this (or other) frame, it is checked by the receiving data link layer if at all it is a duplicate, just the same as in the previous protocol PAR[17]. If it happens to be the expected frame, it is passed on to the network layer and the receiver's window slides up. The number of the last frame received without error is contained in the acknowledgement field. If this number matches with the sequence number of the frame that the sender has been trying to send, the sender comes to knows about its success with the help of the frame stored in buffer and the sender can fetch the next packet from its network layer. If the sequence number fails to agree, it must keep on trying to send the same frame. On receiving a frame, the frame is also sent back [8].

III. PURPOSED PROTOCOL

We proposed a new protocol, Protocol 7 (Priorities Retransmission) to overcome the problems of Selective repeat. It works like Go-Back-N and Selective repeat but it has some changes. It has buffer on both side of receiver and sender. It allows the sender to have more than one outstanding frame at a particular time and receiver to accept out of order frames and store it in its window.

Sender for Priorities Retransmission is modified from that of Selective Repeat. It collects all the NAK which is received from receiver side and store it into an array. If only one NAK is received from receiver side it simply retransmits that frame for which a NAK is received. If more than one NAK is received then find the minimum sequence number of NAK via find_min() and retransmitted that frame, this process is going on until all NAK frames are not retransmitted. It uses optimum bandwidth then Protocol 6.

IV. ALGORITHM FOR PROTOCOL 7

```
find_min(){
for(j=2;j<=nak_count;j++){
for(k=2;k<=nak_count;k++){
```

```
if(min_seq>arr[k])
min_seq=arr[k]; }
send_frame(data,min_seq,frame_expected,out_buf); } }
while (true){
wait_for_event(&event); /*five possibilities: frame_arrival,
cksum_err, timeout, network_layer_ready, ack_timeout */
switch(event){
case network_layer_ready: /*accept,save,& send new frame */
break;
case frame_arrival: /* frame arrived*/
from_physical_layer(&r); /* fetch incoming frame */
if(r.kind==data) /*an undamaged frame arrived */
if((r.kind==nak)&&between(ack_expected,(r.ack+1) %(MAX
_SEQ+1),next_frame_to_send))
{ int min_seq=1024; nak_count=0;
for(k=0;k<NR_BUFS;k++){
wait_for_event(&event)
if(r.kind==nak){
arr[nak_count]=r.seq;
nak_count++; } }
if(nak_count==1)
send_frame(data,(r.ack+1)%(MAX_SEQ+1), frame_expected,
out_buf);
else{
for(i=0;i<nak_count;i++){
find_min(); }
case cksum_err:
if (no_nak) { /*damaged frame*/
send frame(nak, 0, frame_expected, out_buf);
/* nak frame with tn */
} break;
case timeout:
send_frame(data, oldest_frame, frame_expected, out_buf);
/*we timed out*/
case ack_timeout:
send_frame(ack, 0, frame_expected, out_buf);
} /*ack timer expired; send ack*/
```

V. WORKING

The working of new protocol is same as selective repeat protocol. The changes are as follow:

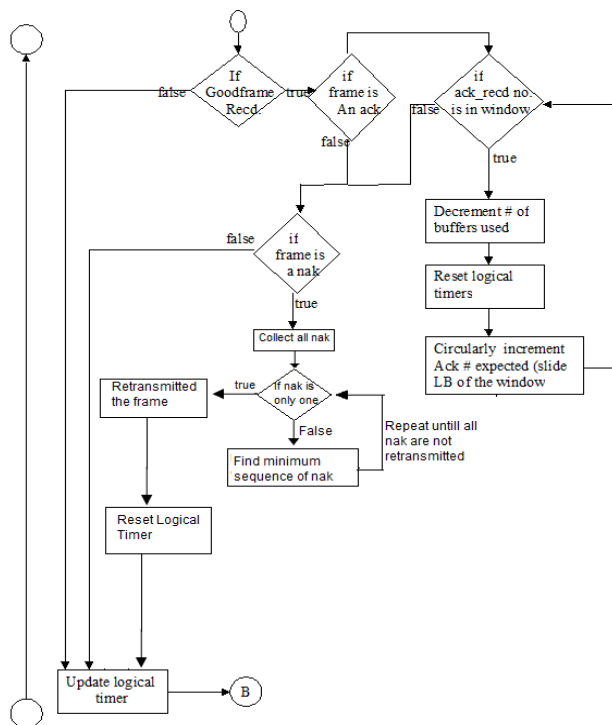


Fig. 1: Working of Protocol 7

VI. PERFORMANCE

Sample Test Result 1:
We have chosen 10% as error rate, and kept the Event is 10000. Here we change the Time out value and calculate the number of frames sent.

TABLE I
TABLE OF TIMEOUT VALUE & NO. OF FRAMES SENT

S No.	Error Rate	Timeout Value	Number of frames sent
01	10%	10	3337
02	10%	30	1281
03	10%	50	785
04	10%	70	564
05	10%	90	440

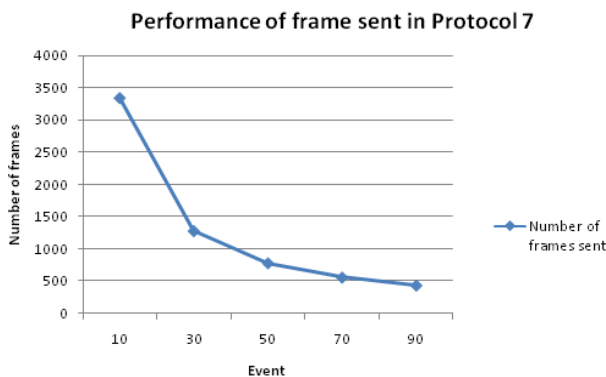


Fig. 2: No. of Frames sent in Protocol 7 when error rate is fixed and timeout value is increases

Sample Test Result 2:
We have chosen 10 as a Time out, and kept the Error rate is 10%. Here we change the Number of Events and calculate the number of frames sent.

TABLE II
Table of Events & No. of frames sent

S. No.	Timeout Value	Error Rate	Events	Number of frames sent
01	10	10%	00	00
02	10	10%	50	18
03	10	10%	100	36
04	10	10%	150	51
05	10	10%	200	67
06	10	10%	250	86
07	10	10%	300	103
08	10	10%	350	121
09	10	10%	400	138
10	10	10%	450	155
11	10	10%	500	164

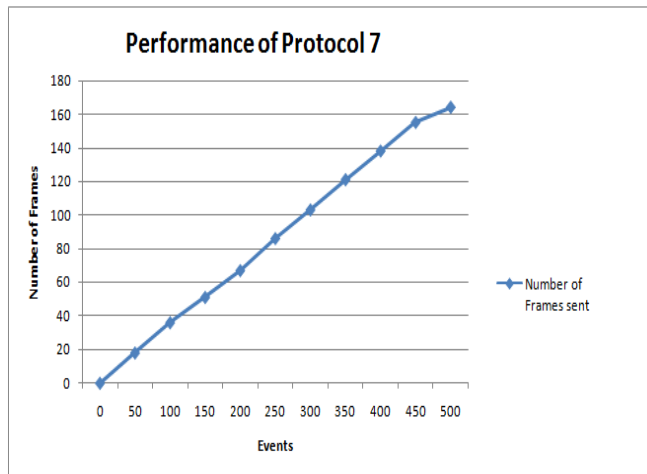


Fig. 3: Number of frames sent when event increases.

VII. CONCLUSION

In this research paper we have briefly explained about the DATA LINK LAYER and its protocols. Data link layer categorized in two parts i.e. Media Access Control and Data Link Control Layer. We work on Data Link Control Protocol which contains six protocols: two for noiseless channel (UTOPIA & Stop and Wait) and four for noisy channels (PAR, Stop-and-Wait, Go Back N, Selective Repeat). UTOPIA is simple but it has the problem of flooding. Stop and Wait protocol is easy to implement but it is not efficient because it needs an error free communication channel which is not possible in present scenario. PAR was easy to implement and it worked very reliably. It was a good trade-off between the ease of implementation and the loss in Bandwidth i.e. the number of frames retransmitted. Sliding window protocols have bidirectional and it work in $(2^m - 1)$ model. It is of three

types: First is One Bit Sliding Window having same characteristics like stop & wait protocol. Second is Go Back N protocol required more input in terms of implementation due to buffer maintenance and keeping sender and receiver in synchronization. This protocol is not efficient due to the large number of frames retransmission. Whenever any frame is lost instead of retransmitting that frame it retransmits all subsequent frames and it also wastes Bandwidth. And third is Selective Repeat it is little improvement over Go Back N and produced far better results in terms of the no. of frames retransmissions, it retransmits only one frame instead of the entire series. But there is no efficient manner of frame retransmission. In real time application like Video conferencing and live telecast, it creates the problem of synchronization. This problem is overcome by the Protocol 7 (priorities retransmission). In this protocol we have priority on the NAK in which we are retransmitting the frame according to minimum sequence number of NAK. So Protocol 7 gives optimum performance than protocol 6.

REFERENCES

- [1] Behrouz A Forozan, "Data Communications and Networking", Fourth Edition.
- [2] H. Zimmermann, "OSI Reference Model - The ISO Model of Architecture for Open Systems Interconnection", IEEE trans. On Communications, Vol COM-28, pp 425-432, April 1980.
- [3] Steven Cherry, "A Broadband Utopia", IEEE spectrum, May 2006
- [4] S Chen, "ELEC3030(EL336)", Computer Networks, Electronics and Computer Science, University of Southampton.
- [5] F. Argenti, G. Benelli and A. Garzelli "Generalised Stop and Wait Protocol", Electronic Letters, Vol.28, No. 9, 23rd April 1992.
- [6] HOLZMANN, G.J. "Design and validation of computer protocols" Prentice-Hall, Englewood Cliffs, NJ, 1991
- [7] W. STALLINGS "Data and Computer Communications", Prentice Hall Upper Saddle River, NJ, 1997, 5th edition.
- [8] Andrew S. TANENBAUM, "Computer Networks", Prentice Hall, Upper saddle River, NJ, 1996, 3rd edition.
- [9] D. Hercog, "Generalised Sliding Window Protocol", Electronic Letters, Vol. 38, No. 18, 29 Aug 2002.
- [10] G. ogasawara, t.ju, s.kota, "Experiments with Tactical network simulation, routing and management", milcom '96, confrence proceedings, ieeee, volume 2.
- [11] LIN, S., COSTELLO, JR, D., and MILLER, "Automatic-repeat request error-control schemes", IEEE Commun., 22 December 1984.
- [12] LINs and COSTELLO, JR, I "Error control coding: fundamentals and applications" (Prentice Hall, Englewood Cliffs, New Jersey, 1983).
- [13] M. E. Anagnostou and E. N. Protonotatos, "Performance analysis of the Selective Repeat ARQ protocol," IEEE Trcnls. Qmmun., vol. COM-34, Feb. 1986.
- [14] K. Brayer and S. Natarajan, "An investigation of ARQ and hybrid FECARQ on an experimental high latitude meteor burst channel," IEEE Trans. Commun., vol. 37, Nov. 1989.
- [15] O. C. M. B. Duarte and H. M. Lima, "A new selective repeat scheme Actual Environmental Performance Analysis Conference. June 1988.
- [16] F. Halsall, Data Communications, Computer Networks and Open Systems-Addison-Wesley, 1992.
- [17] J. J. Metzner and D. Chang, "Efficient selective repeat ARQ strategies for very noisy and fluctuating channels," IEEE Trans. Commun., vol. COM-33, May 1985.
- [18] M. J. Miller and S. Lin, "The analysis of some Selective Repeat ARQ schemes with finite receiver buffer," IEEE Trans. Commun., vol. COM-29, Sept. 1981.