International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2018 IJSRCSEIT | Volume 3 | Issue 3 | ISSN : 2456-3307

Low Latency Automatic Repeat Request Protocol for Time Sensitive GSM-Enabled Smart Phone Video Streaming Services

Vincent Omollo Nyangaresi [1], Dr. Silvance O. Abeka [2], Rebecca N. Arika [3]

[1,2] School of Informatics & Innovative Systems, Jaramogi Oginga Odinga University of Science & Technology, Kenya

[3]School of Computing and Information Technology, Jomo Kenyatta University of Agriculture & Technology, Kenya

ABSTRACT

Automatic repeat request protocols (ARQ) are fundamental error control techniques employed in wired as well as wireless networks. These protocols facilitate error recovery based on feedback messages and retransmissions. However, the performance of these protocols is heavily influenced by the propagation delay occasioned by the requirement that the sender must wait for the receiver feedback before making a decision on whether to retransmit a packet or shift to the right of the sliding window and transmit another packet. As such, the performance of ARQ protocols heavily relies on the feedback channel reliability. Considering channels with long round trip times, the consequence is that frame delivery may be severely belated. In video streaming on smart phones, delays are undesirable especially when users are watching live events. Such applications can therefore be regarded as time sensitive. The weaknesses of the current ARQ protocol include inefficiency in bandwidth usage due to retransmissions of frames already sent and received correctly when one of the frames is lost, time wastage by the stop and wait behaviour. Based on these shortcomings, towards the end of this paper, dual asynchronous prioritized retransmission ARQ (DAPR-ARQ) protocol is suggested as a possible remedy for these ARQ shortcomings.

Keywords*:* GBN-ARQ, SAW-ARQ, SR-ARQ, time sensitivity, video streaming

I. INTRODUCTION

Reliable communication protocols are the ones that provide some guarantee that the sent frames will get to their destination correctly, or if this fails, then they inform the layer above them that frame delivery has failed [1].For this to happen, reliable protocols require a bi-directional communication channel so that the receiver can transmit short acknowledgement packets back to the sender. These acknowledgements serve to inform the frame sender the status of the delivery, whether it was successful or not.

Here, if a frame successfully arrives at the destination, then the receiver sends back a positive acknowledgement (ACK). On the other hand, if a frame fails to arrive, or arrives in error, the destination machine sends back a negative acknowledgement (NACK) which essentially triggers a frame retransmission. This process is known as Automatic Repeat request (ARQ). The

implementations of ARQs diverge in the way the frame source and destination interact during the frame transmission process. The classic ARQ schemes are Stop-and-Wait (SW), Go-Back-N (GBN), and Selective Repeat (SR),sorted by increasing throughput efficiency, as well as complexity cost.

In their paper, [2] explain that future wireless networks envisage ultra-reliable communication that feature proficient utilization of the limited channel resources. In closed-loop repetition protocols, retransmission of packets is facilitated using a feedback channel. Protocols such as ARQ are employed in wireless technologies to provide the communication channel with diminished rate of packet outage and increased average throughput.

In [3], it is noted that Netflix and YouTube utilize reliable transport protocols in the provision of data transport for time sensitive applications such as video streaming services exampled. For the case of fixed wired networks where the packet erasure rate is low, the quality of service (QoS) for these applications is usually satisfactory. However, the adoption of wireless technologies creates non-congestion related packet erasures within the network that can leads to degraded TCP performance and unacceptable QoS for time sensitive applications.

The major causes of the degraded performance are TCP congestion control and head-of-line blocking (HoL) when recovering from packet losses is another. HoL happen when a number of packets are held up by the first packet and is occasioned by using techniques such as selective repeat automatic-repeat-request (SR-ARQ), which is used in most reliable transport protocols such as TCP.

According to [4], the transport layer is charged with the responsibility of providing services such as flow control, reliability, congestion avoidance and multiplexing based on port numbers. The main transport layer protocols are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). While UDP is a simple unreliable protocol that sends independent datagrams without expecting a response, TCP is a connection-oriented protocol. This means that TCP facilitates a dependable stream of data with definite order of messages. In addition, TCP includes flow control and congestion avoidance. However, due to the extra overhead associated with the TCP connection and its various additional services, it is heavier than UDP. On its part, UDP has low overhead associated with it and provides unreliable but quick data transmission.

In TCP's flow control, the idea is to determine the data transmission rate to avert congestion at the receiver. Congestion crops up when two or more terminals are communicating, and the sender transmits faster than the receiver can accept and process. This effectively causes the receiver's buffer to fill up and overflow. This normally happen when the receiver is subjected to heavy traffic load or due to its less processing power compared with that of the transmitter. An overflowing buffer leads to the loss of new frames as the receiver is still processing old ones. To avert this flow control is implemented, in which the receiver has to determine the rate of transmission when communication is initiated.

The open system interconnection (OSI) data link layer serves two main purposes namely data link control and media access control. Here, link control is concerned with the design and procedures for communication between two adjacent nodes, with crucial responsibilities being flow control and error control, which are collectively referred to as data link control. According to [5], the two fundamental approaches to error control in digital communications are forward error correction (FEC) and automatic repeat request (ARQ).

In FEC systems, parity-check bits are appended to each sent frame to form a code word depending on the error-correcting code that is being utilized. Essentially, the receiver tries to locate and rectify the detected errors in the received frame, after which the

decoded data block is delivered to the end user. In this scenario, a decoding error takes place when the output of the decoder is a different codeword from the one that was originally sent at the source. Essentially, FEC systems were meant for simplex channels where information flows in one direction.

In their study, [6] explain that in communication networks, the major error control mechanisms employed on a per-hop basis to conserve data integrity are FEC and ARQ. While FEC schemes operate at the physical layer, ARQ techniques operate at the data link layer. In FEC an error, correcting code is utilized to generate the parity check bits. When an error is detected, the receiver traces and corrects the errors in the received packet.

In [7], the authors explain that flow control schemes are of two classes; those which provide feedback (ARQ),and those which do not send any kind of feedback (forward error correction-FEC). In FEC, the sender appends redundancy to its messages to permit the receiver to detect and correct errors without asking the sender for additional data. In this way, retransmission of data is thwarted but at the expense of higher bandwidth requirements. In essence, FEC provides one-way connection between the sender and the receiver without feedback information. It is insensitive to channel error ratio, although it has lower throughput in high signal to noise ratio (SNR) for a fixed coding rate than ARQ. Moreover, throughput can demean due to redundancy and decoder complexity [9].

In an ARQ scheme, a high-rate error-detecting code coupled with some retransmission protocol is utilized. Unlike the FEC systems, ARQ protocols require feedback channel. Here, incorrectly received packets are retransmitted until they are detected as error-free or until they reached the maximum allowable number of transmissions [10]. This scheme employs cyclic redundancy check (CRC), due to its ease of implementation, for error detection and the destination machine send either acknowledgement or negative acknowledgement to the source machine based on the CRC detection result. Using this scheme, higher throughput in high SNR regions can be achieved compared to FEC scheme, but a lower throughput in intermediate SNR region.

In addition, it is sensitive to channel error ratio under severe fading conditions. In their study [11] discuss that error control in data link layer is concerned with error detection and error correction. This means that when data is lost or damaged during the transmission, error control will inform the sender, specify the frames and ask retransmit, a process referred to as automatic repeat request.

II. REVIEW OF THE CURRENT ARQ PROTOCOLS

ARQ protocols operate by appending parity or some redundant bits to the transmitted data stream which are then employed by the receiver to detect an error in the received data. Here, the receiver makes no effort to correct the errors but instead it sends a NACK, prompting the source machine to retransmit the erroneous frame [12]. ARQ protocols are distinguished by their persistency, which is the enthusiasm of the protocol to resend lost frames to ensure dependable delivery of traffic across the communication channel. Fundamentally, there are four ARQ schemes, namely Stop-and-wait ARQ (SAW-ARQ), Go-Back-N ARQ (GBN-ARQ), Selective Repeat ARQ (SR-ARQ) and Hybrid ARQ (HARQ) as discussed in the subsection that follows.

A. Stop and Wait ARQ

In this protocol, a frame is transmitted by the source machine to the receiver, and then the sender waits for a confirmation of correct receipt after which another frame for the next packet is sent. However, if the receiver sends a NACK, then the sender will resend the indicated frame. To inform the receiver the packet layer to which the frame belongs, a serial

number is employed to uniquely identify network layer packets [13]. The strength of this protocol is that it is easy to implement but it is a low efficiency mechanism because the channel utilization is low.

As shown in Figure 1, when packet 0 *(pkt 0)* is sent, the receiver responds with Acknowledgement 0 *(ack 0)*, meaning that this packet has been successfully received. Similarly, when *pct 1* is sent, the receiver responds with *ack 1*, meaning that packet 1 has been detected without errors. As [14] points out, a timer is also utilized in case packet or acknowledgment is lost during transmission. Here, after transmitting a packet, the sender waits for the ACK or

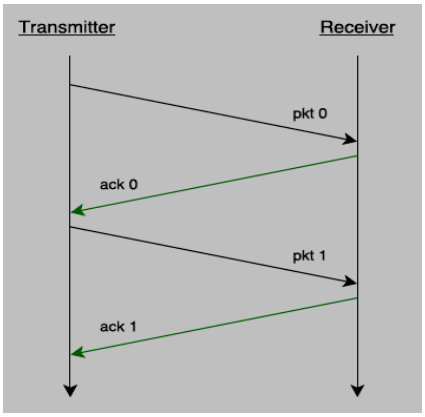

Figure 1. Stop and Wait Protocol

NACK until for some specified time. When it does not receive the ACK or NACK within that time, a timeout takes place and the packet is retransmitted as shown in Figure 2.

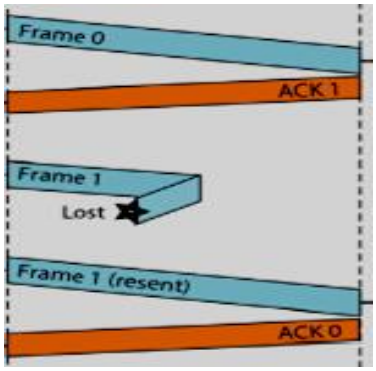

Figure 2.SAW with Timeout

As Figure 2 demonstrates, frame 0 is transmitted and acknowledged by ACK 1, meaning that it was successfully received. However, during the transmission of frame 1, a timeout occurs and it is lost. As such, frame 1 is resent and acknowledged by ACK 0, meaning that now it has been detected successfully. Unfortunately, this protocol of flow control involves a large amount of waiting due to the requirement that the sender wait for the receiver ACK or NACK after each packet is sent [15]. Consequently, the transmission will often be slow and inefficient.

B. Go Back N ARQ

In this protocol, the sender transmits packets to the receiver continuously, receiving acknowledgments as well. Here, when the receiver sends a NACK to the sender, the negatively acknowledged packet is retransmitted immediately and all already-transmitted packets following it [16]. One of the challenges of SAW ARQ is its low utilization of the communication channel. Go-Back-N (GBN) overcomes this limitation by keeping the channel busy while the sender awaits acknowledgments for the sent frames.

The idea behind GBN is that the sender transmits the first frame and continues sending frames as it awaits the acknowledgement of the first frame from the destination machine. When the first frame is detected correctly at the receiver side and its acknowledgement is sent back to the sender and received correctly, then the transmitter is completely done with this frame. While the acknowledgement of the first frame is going on, the handling of subsequent frames is going on simultaneously.

In their paper, [17] noted that in GBN, piggyback is employed to interleave the acknowledgements in the headers of the information frames. In piggyback, the control information is conveyed to the other end along with the data itself, improving efficiency. A techniques referred to as pipelining is also utilized in which the transmitter continuously sends frames for a time equal to the frame round trip transit time and

it does not require the source machine to wait for an ACK before transmitting the next frame. In case a frame in the middle of this stream of frames is damaged or lost, the receiver merely discards all subsequent frames, sending NACKs. The consequence is that the receiver refuses to admit any frame except the next one it must forward to the network layer.

When viewed from the sliding window perspective, GBN involves sending a certain number of frames continuously as specified by a window size, even without the receiver replying by sending an ACK packet. According to [18], this is a special case of the broad sliding window protocol with a transmit window size of N, where the receive window size is a unity. Here, the communicating peer devices exchange N frames without any ACK and the receiver is tasked with the responsibility of keeping track of the sequence number (SN) of the next expected frame to be received.

It therefore sends this SN as part of every ACK it sends and any frame that does not have the expected SN is discarded by the receiver. This can be either an already ACKed duplicate frame or an out-of-order frame it expects to receive later. As such, it is only the last correct in- order frame that will be ACKed.

Once all the frames in the sender's output buffer window are transmitted, the sender will notice that all the sent frames in that window are outstanding, beginning with the first frame that was lost [19]. It will hence regress to the last acknowledgement it received in order to obtain the sequence number of the next expected frame. Thereafter, it starts filling its window with this frame and the communication process will continue once more as shown in Figure 3.

As shown in this figure 3, the window size in this case was two frames, allowing frame 0 and frame 1 to be sent before an acknowledgement could be received.

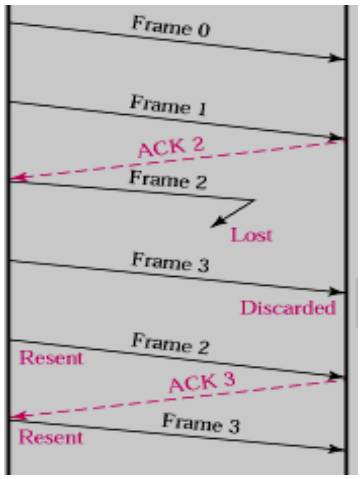

Figure 3.GBN Protocol with Timeout

The first acknowledgement *ACK2* is an accumulative acknowledgment for frame 0 and frame 1. During the transmission of frame 2, a timeout occurs and is lost but the transmission of frame 3 was successful. However, although frame 3 went through, it is discarded since it is does not have the expected SN. Therefore, both frame 2 and frame 3 are resent.

The receiver keeps track of the SN of the next frame it expects to admit, and sends this SN with every ACK it sends. If a frame from the sender does not reach the receiver, this destination machine has to terminate acknowledging preceding received frames. When the retransmission timer expires, the sender retransmits all unacknowledged frames in order, starting with the lost or damaged frame.

Unfortunately, during the transmission process, if a given frame encounters an error in detection at the destination machine, this frame is ignored together with any other frame that came after it. Upon the completion of transmission process, the sender has to retransmit the frames starting from the one that encountered an error, hence the name GO-BACK-N where N stands for the number of the frame that encountered detection error [20]. The strength of this protocol is its higher transmission efficiency since as long as negative acknowledgement is not received, the transmitter will continue sending until the send sliding window is emptied.

As [21] noted, the GBN ARQ is more efficient than SAW ARQ, since waiting for an ACK for each sent packet is not an obligation and the packets exchange still goes on as ACKs are being sent. This means that the communication link is still being utilized. Consequently, the number of packets that are sent is increased since the waiting time is done away with.

On the flip side, this protocol requires a sophisticated storage at the destination terminal to buffer all the sent data. Consequently, it is well suited for systems with low data transfer rate and short round trip time. In addition, in GBN, lost or damaged frames or failing to acknowledge reception of frames by the receiver results in the retransmission of all frames in the windows, even multiple times, including the error free ones. To overcome this problem, Selective Repeat ARQ (SR-ARQ) is employed as discussed in the section that follows.

C. Selective Repeat AQR

This protocol is more efficient than GBN since only frames that received NACK code words are retransmitted. Here, the sender transmits frames continuously until a NACK arrives at the transmitter [22]. When this happens, the sender retransmits the NACKed frames without resending the transmitted packets following it the sliding window as shown in Figure 4.

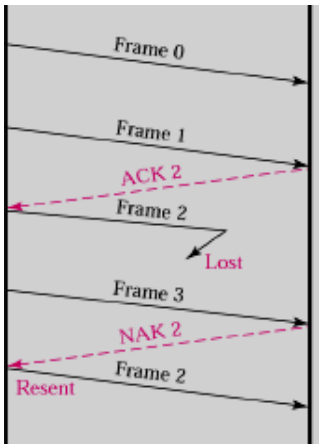

Figure 4.SR-ARQ with Timeout

Figure 4 demonstrates that the window size was two frames, allowing the transmission of both frame 0 and frame 1 before receiving an acknowledgment. As such, ACK 2 is an accumulative acknowledgement for both frame 0 and frame 1. During the transfer of frame 2, a timeout is encountered and is lost. However, the transmission of frame 3 went through. A NAK 2 is generated at the receiver, meaning that frame 2 has to be resent. Unlike the case for GBN where both frame 2 and frame 3 were retransmitted, in SR-ARQ, only the lost frame 2 is retransmitted.

This requires that the receiver maintain a sequencing buffer that is aimed at preserving the original arriving order of packets. In GBN protocol, the original order of the code words is automatically preserved. However, SR-ARQ has to maintain buffer space to accumulate the properly received code words that cannot yet be released. The timer is set for all the outstanding windows and the frame whose timer is expired is resent.

According to [20], the GBN protocol is not efficient in communication channels with high error rates since it has to retransmit the erroneous frame and all the other frames after it. The SR-ARQ addresses this setback by retransmitting only the NACKed frames and is the most effective of the ARQ protocols. Its strengths are that it improves time and channel utilization.

This protocol improves GBN by maintaining buffers on both the sending and receiving sides, which permits the transmitter to have more than one outstanding frame at a time and receiver to accept out of order frames and store them in its window [23]. Since the receiver maintains a window of frames, only the timed out frame needs to be resent and not the whole series.

As was the case with GBN protocol, the sender transmits a number of frames as dictated by the window size, even without the necessity of waiting for individual ACKs from the receiver [24]. It differs from GBN in that the receiver can selectively rebuff

a single frame when it is corrupted, which is then retransmitted alone. Instead of dropping out-of-order frames, the receiver admits and buffers them.

The setback of SR-QRA is that it is difficult to implement. There is also the requirement that the SNs be greater than the window size so that no overlap can occur in the window. This permits the receiver and sender to be synchronized even when frames and ACKs are lost at very high rates [25]. The buffering and ACKs permit this protocol to handle congestion, damaged frames and lost frames efficiently.

D. Hybrid ARQ

This protocol works by combining FEC and ARQ and achieves enhanced throughput and efficiency by thwarting the decoding complexity of FEC and time delay of ARQ. This renders it more suitable of high-speed wireless access system [26].Here, a message is encoded using FEC and then broken down into several blocks which are thereafter transmitted one after another until the receiver decodes the entire message successfully. As [27] discuss, there are three variants of HARQ, namely Type I, Type II and Type III.

In Type I HARQ, there is always error detection code in addition to FEC encoded data and FEC code is first decoded at the destination machine. In case errors are detected, the sender is requested to resend the affected frame and the erroneous packet is discarded. In Type II, the damaged frame is not rejected but resent along with some incremental redundancy implemented by the sender for subsequent decoding. On its part, Type III works in a similar manner as Type II, only that here each packet is self-decodable.

In their study, [28] point out that based on which part of the original codeword is be used for retransmissions with soft combining, the HARQ protocol can be broken down into three categories, namely full incremental redundancy-based HARQ (FIR-HARQ), Chase combining-based HARQ (CC-HARQ), and partial incremental redundancy-based HARQ (PIR-HARQ). In FIR-HARQ, fresh parity bits that have not been transmitted up to the prior transmission are sent without systematic part utilization for each retransmission. As such, it can attain coding gain offered by parity bits. This protocol can be regarded as Type-II HARQ process without self-decodability. Its setback is that in situations where the systematic part is seriously damaged or corrupted due to deep fading condition, it is unfeasible to recover it by retransmissions. In addition, FIR-HARQ is complicated to realize.

In CC-HARQ, all the coded bits for the first transmission are reused for retransmissions and is a special case of Type-III HARQ with one redundancy. For this reason, it is referred to as repetition time diversity scheme. Its strengths are that it can obtain SNR gain by simple hardware implementation and it requires smaller buffer size in a receiver than PIR-HARQ and FIR-HARQ. On its part, PIR-HARQ can be regarded as a Type-III HARQ process in which part of the coded bits for the early transmission, mostly the systematic part is employed for retransmissions while a number of extra parity bits are freshly transmitted for each retransmission. Its strengths are that it can attain both SNR and coding gains.

HARQ facilitates consistent communications without the knowledge of channel state information and is therefore an fundamental part of modern communication standards such as 3rd Generation Partnership Project (3GPP) long-term evolution (LTE) and LTE advanced (LTE-A).

III. RELATED WORK

In their study, [18] proposed multiple SAW processes that are executed in parallel so as to compensate for the idle time. This scheme has the potential of

improving throughput efficiency. This scheme differs from the original SAW-ARQ in that whereas packets always arrive in sequence, in N-SAW packets may arrive at the out of sequence at the destination machine. This is because although at the beginning of data transmission frames are sent in sequence, when errors occur, the diverse SAW connections may need varied number of packet retransmissions. This effectively results in out of sequence arrival of packets. The demerit of this scheme is that it requires a reordering buffer at the receiver to re-sequence the frames and deliver them in-sequence to the higher layer.

Another study by [22] suggested a juggling-like stop-and-wait (JSW) transmission scheme that facilitated the utilization of continuous ARQ protocols over half-duplex underwater acoustic links. Cognitive radio (CR) has also been proposed as a mechanism for improving spectrum usage. In [16], a Cognitive Selective Repeat HARQ (CSR-HARQ) scheme is suggested for a spectrum overlay environment. Here, it is assumed that primary user (PU) sends information based on time-slots. During a given timeslot, the CR sender first listens to the PU channel and if a free time slot is established, a number of packets are sent to the cognitive users (CU) receivers based on SR-HARQ. It was noted this scheme led improved throughput and delay performance. The challenge is that it requires re-sequencing buffer for queuing the out-of-order error-free packets.

In [29] cognitive GBN-HARQ (CGBN-HARQ) is suggested. This is a CR communication scheme that enables a cognitive user to determine the activity of the PUs over a primary radio (PR) channel and if found to be free from Pus, the CR sends data using the modified Go-Back-N HARQ (GBN-HARQ) protocol. Here, the activity of PUs on the PR channel is modeled as a two-state Markov chain with ON and OFF states. On the flip side, the CU may mistakenly detect the ON/OFF activity of the PUs in the channel, hence resulting in false-alarm or misdetection.

Reducing the in-order delivery delay of reliable transport layer protocols over error prone networks has been observed to enhance application layer performance, more so for applications that have time sensitive constraints such as streaming services [3]. Systematic random linear network code (RLNC) coupled with a coded generalization of SR-ARQ was established to reduce the time required to recover from losses [30]. This was realized by injecting coded packets at crucial locations to the original data stream. By overcoming packet losses and limiting the number of required retransmissions, delays were reduced. The only challenge is that correlated losses or erroneous information about the network leads to the receiver's inability to carry out the decoding.

A study by [31] evaluated the delivery delay, which consists of transmission and re-sequencing delays of the SR-ARQ operating over the JSW scheme. The results indicated that this scheme is immune to the round-trip delay, making it suitable for terrestrial communications. Here, the total delay of an ARQ protocol was attributed to three components namely, queuing delay, transmission delay and re-sequencing delay. In this case, queuing delay refers to the duration from the time a packet arrives at the sender until its first transmission attempt and this is correlated to the channel characteristics and the packet arrival process.

On its part, the transmission delay is the time from a frame's first transmission until its successful arrival at the receiver and this included all retransmission delays. This delay is only influenced by the channel characteristics. On the other hand, re-sequencing delay refers to the waiting time of the packet in the receiver re-sequencing buffer. It is delay experienced by a frame from the time of its correct reception up to its in-order delivery to the upper layers.

In [32], the authors proposed a two-user cooperative automatic repeat request (C-ARQ) protocol that combines cooperative diversity at the physical layer and ARQ at the link layer. C-ARQ result when ARQ protocols are employed in cooperative diversity system to attain higher link reliability. In essence, C-ARQ allows terminals other than the sender and the receiver to actively aid in the delivery of data. In this scenario, distributed Alamouti space-time block coding (DASTBC) was employed to achieve cooperative diversity while ARQ was utilized to improve the data link layer reliability.

The results showed that the performance gain of average frame error rate with ARQ retransmission was about 2 dB more than the case without ARQ retransmission in the same conditions. In addition, [33] suggested a coordinated hybrid automatic repeat request (HARQ) approach which can enhance network outage probability and the users' fairness.

IV. CRITIQUE OF CURRENT ARQ PROTOCOLS

One of the greatest challenges of the current ARQ protocols is that their performance is heavily dependent on the feedback channel reliability and as such, if an ACK is lost, the ACKed frame will be resent although it was detected correctly at the receiver side. The purpose of the feedback channel is to limit repetitions to only when the initial frame delivery fails and hence boosting data channel efficiency. However, inevitable feedback channel impairments may cause unreliability in packet delivery. The second challenge of ARQ protocols is that a decoding failure report, in form of NACK erroneously detected as ACK results in detrimental packet outage.

Attempts have been made to increase feedback reliability by use of repetition coding. However, this mechanism is expensive to the receiver and erroneous feedback detection leads to an increased packet delivery latency, diminished throughput and reliability key performance indicators. In the latest releases of LTE, blind HARQ retransmissions of frames has been suggested as a solution for reducing feedback complexity of broadcast HARQ and boosting reliability. However, despite its reduced simplicity, resource utilization efficiency of the system can severely be diminished. This is because typically a high number of transmissions is effectively decoded in the initial attempt in distinctive link adaptation configurations.

Another setback of ARQ protocols are head-of-line blocking issues that come about as a result of using techniques such as SR-ARQ. In spite of the fact that SR-ARQ ensures high efficiency, its setback is that packet recovery due to losses can take a round-trip time (RTT) or more. In ideal situations, the RTT or the bandwidth-delay product is very petite and the feedback is nearly instantaneous.

On these conditions, the SR-ARQ offers near optimal in-order delivery delay. Unfortunately, due to queuing and great physical distances between a client and server, feedbacks are normally belated. This negatively impact applications that need reliable delivery with constraints on time between the transmission and in-order delivery of a packet. Although SR-ARQ is the most efficient ARQ protocol, it is infeasible in underwater acoustic communications due to the half-duplex feature of typical underwater acoustic modems.

In addition, although SR-ARQ is one of the most efficient error control protocols for packet-switched communications over lossy channels in terms of throughput and delivery delay, it cannot ensure that packets are in-order delivered. This means that it cannot preserve the order of packets at the receiver. Due to this, all the upper-layer protocols that require in-order packet delivery cannot process the already received information frames. As a result, all the successfully received frames have to be stored in a buffer, referred to as re-sequencing buffer, until the

entire original frame stream can be passed to the upper layers as per the original order.

V. ARQ PROTOCOLS EVALUATIONS

A number of performance measures have been developed for ARQ protocols which include throughput delay and energy efficiency [29]. In addition, during the evaluation of various ARQs based on efficiency, Bit Error Rate (BER) has been noted to be one factor that influences this efficiency [25]. As such, BER can be computed for different window sizes in order to figure out optimum value of window size for range of BER.

In their paper, [22] point out that the performance evaluation of the three classical ARQ protocols for multiple identical channels can be carried out using the mean number of packets effectively transmitted per unit time, and the mean transmission delay, which is the average time between the instant when a packet is transmitted for the first time and the instant when it is successfully received. Another study by [16] analysed the throughput of ARQ protocols using average packet delay, and end-to-end packets delay of the CSR-HARQ protocol.

VI. METHODOLOGY

In this paper, three performance parameters namely end to end packet delay and throughput. In this paper, throughput was taken to be the number of frames that were successfully delivered to the receiver for each of the ARQ protocols. For the case of SAW, the frames to be transmitted were selected and then the influence of transmission timeouts on throughput was observed. In addition, the impact of the channel bandwidth, which was taken to be the speed of frame delivery over the communication channel, on throughput was investigated. In GBN-ARQ and SR-ARQ protocols, apart from end to end delay and throughput, an extra parameter, called the window size was employed. As was the case for SAW, the influences of window size, timeouts and channel bandwidth on throughput were investigated.

The three ARQ protocols were simulated in this paper and hence the first task was the design of a GUI where selection of a particular flow control protocol would be selected and investigated. The resulting interface for the simulation environment is shown in Figure 5.

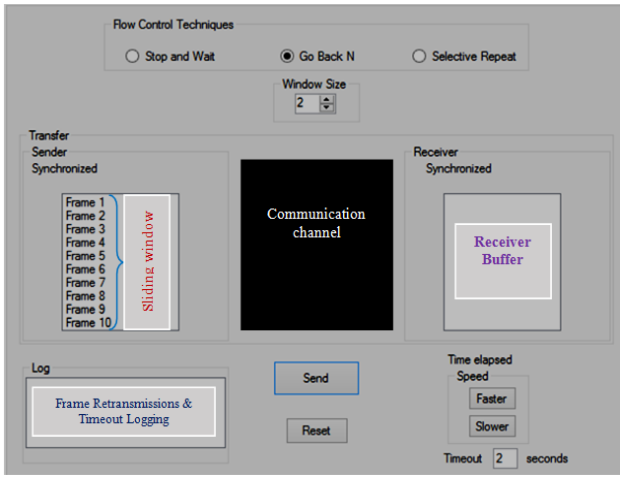

Figure 5. ARQ Simulation Interface

The figure shows that the first section of this interface was that for the selection of the flow control protocol which were SAW, GBN and SR-ARQ. The second section was for the choice of the window size for the case of GBN and SR-ARQ. For SAW, the window size is always unity (1) since only one frame is sent and the ACK waited for before sending the next frame, and hence cannot accept any other window size. The third section was that of the source, communication channel and destination together with sender sliding window containing the frames to be sent. In this case, ten frames were considered, labelled frame 1 to frame 10.

The fourth section was that of the log section, bandwidth adjustment and time out adjustment. The logs were employed to indicate the frames that were successfully detected at the receiver as well as those ones that were timed out and hence required retransmission. The bandwidth section was employed to vary the transmission speed to study its

impact on frame delivery. Lastly, the timeout was employed by the sender to determine how long to wait for the ACK from the receiver before considering the transmitted frame lost and therefore initiate a retransmission.

For the case of SR-ARQ and GBN ARQ, the communication channel consisted of simultaneous transmission of the frame and acknowledgement as shown in Figure 6.

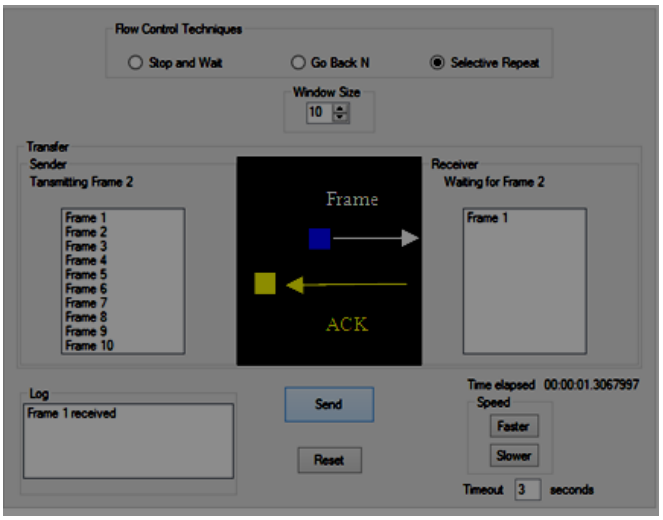

Figure 6.SR ARQ and GBN Full Duplex

While the frames to be transmitted travelled from the left hand side (from the sender) towards the right hand side (towards the receiver), the acknowledgement travelled from the right hand side towards the left hand side (towards the sender) as indicated by the arrows in Figure 6.However, in SAW, the transmission is half-duplex in that the acknowledgement is only sent upon the successful arrival of the frame at the receiver.

Similarly, during the transfer of an acknowledgement, no frame transmission is allowed, hence the name stop and wait. This is in sharp contrast to the GBN and SR-ARQ where the frame transmission can take place simultaneously as the receiver is acknowledging the correctly received frames, hence the name full duplex. The results obtained are discussed in the section that follows.

VII. RESULTS AND DISCUSSIONS

The first ARQ to be investigated was SAW where only one frame was selected for transmission at a time at various bandwidth values and timeouts. To start with, the frame 1 was selected for transmission and the timeout was set to one second. The results obtained are shown in Figure 7.

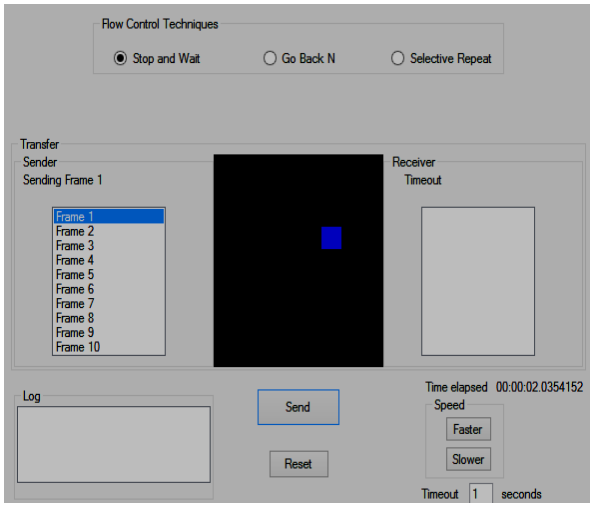

Figure7.SAW with Timeout of 1 Second

The square in the black background was this transmitted frame and as can be seen, the time elapsed was 2.0354152 seconds, which was more than the set timeout value of 1 second. As such, frame 1 was never delivered to the receiver and as shown at the receiver side, it is marked as 'Timeout'. The value of timeout was then incremented to 2 seconds and the outcome obtained is shown in Figure 8.

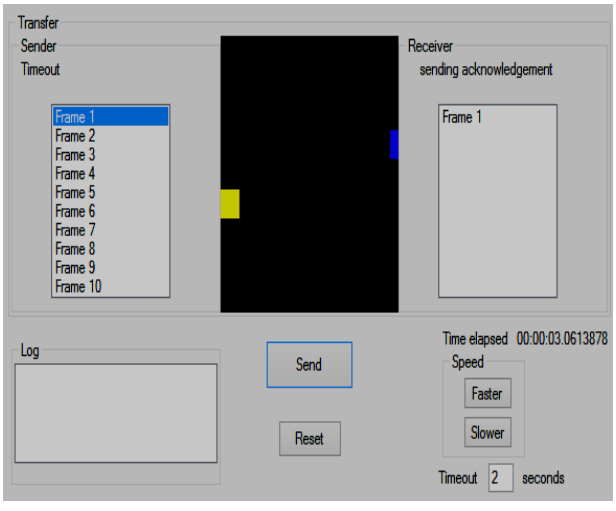

Figure 8.SAW with Timeout of 2 Seconds

It can be seen that with a timeout of 2 seconds, frame 1 was successfully delivered at the receiver after which an acknowledgment (left moving square) for this frame 1 was transmitted to the sender. Unfortunately, this acknowledgment never reached the sender and is labelled 'Timeout' at the sender. The elapsed time was now 3.0613878 seconds. Once again, the timeout was increased to 3 seconds and observation made. Figure 9 shows the results obtained.

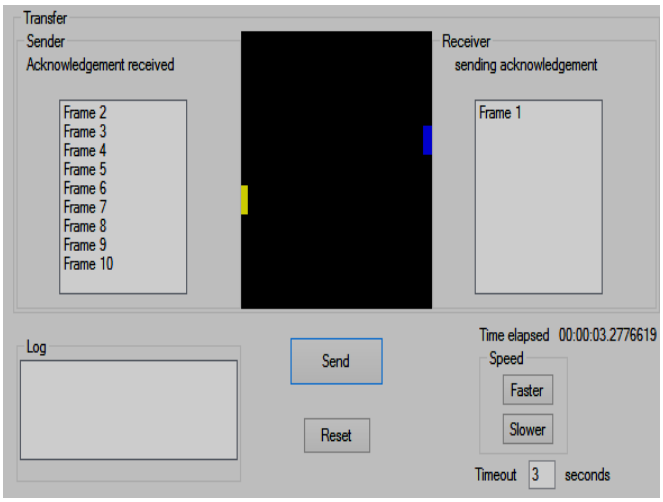

Figure 9.SAW with Timeout of 3 Seconds

With a timeout of 3 seconds, frame 1 was not only delivered successfully at the receiver but its acknowledgement was also successfully delivered at the sender, and the sender labelled this as 'Acknowledgement received'. Thereafter, the sender's sliding window shifts to the right and frame 2 is now a candidate for transmission, having removed frame 1 from the queue.

The simulation results for network bandwidth were investigated by selecting either faster or slower options. When the slower option was selected, a timeout was observed at the receiver, even at a timeout value of 3 seconds as shown in Figure 10. As Figure 10 demonstrates, for faster networks even with a timeout value of 1 second, frame 1 has been successfully delivered at the receiver and the receiver has already dispatched an acknowledgment for this frame although this ACK is now timeout.

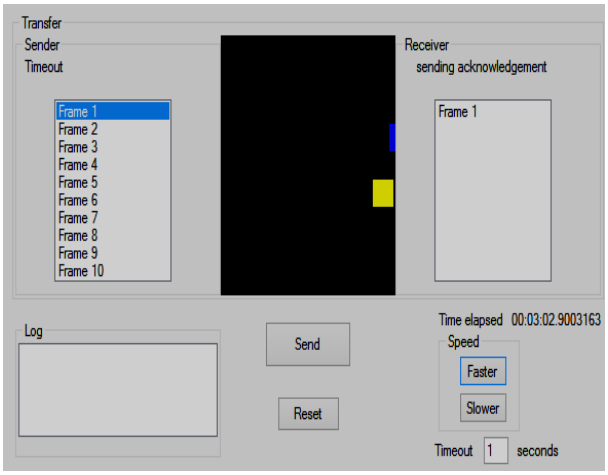

Figure 10.SAW with Timeout of 1 Seconds for Faster Network Bandwidth

This was unlike Figure 7 where frame 1 was timeout before reaching the receiver side. To investigate slow network bandwidth, the timeout was set to 3 seconds and the slower option selected as shown in Figure 11.

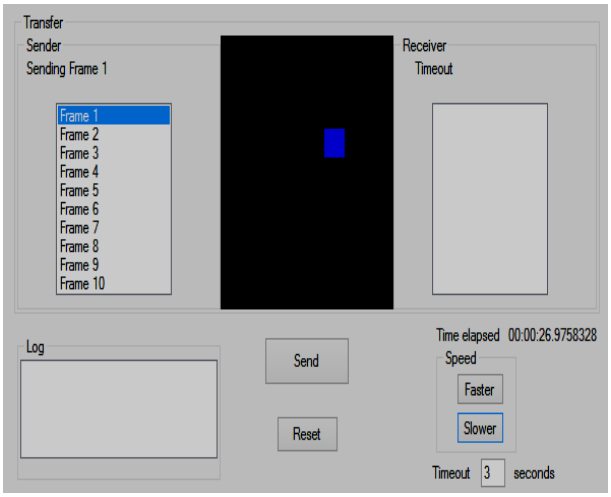

Figure 11.SAW with Timeout of 3 Seconds for Slower Network Bandwidth

Unlike the case in Figure 9 where a timeout of 3 seconds was enough for frame 1 to be successfully delivered at the receiver an its acknowledgment received at the sender, for slower network bandwidth depicted in Figure 11, 3 seconds was not even enough to deliver frame 1 to the receiver. Therefore, apart from sender timeout which may lead to frame loss in the communication channel, the network bandwidth influences the success or failure of the frame transmission process.

To investigate GBN, a window size of two frames was initially selected and a timeout of 1 second was chosen. Figure 12 shows the results obtained. The behaviour of the window size, sender sliding window, receiver buffer and log sections were points of interest in this protocol.

As illustrated in Figure 12, with a window size of 2 frames and a timeout value of 2 seconds, two frames were sent to the receiver, one after another without waiting for any ACK. Unfortunately, as illustrated in the log section, both frame 1 and frame 2 timed out.

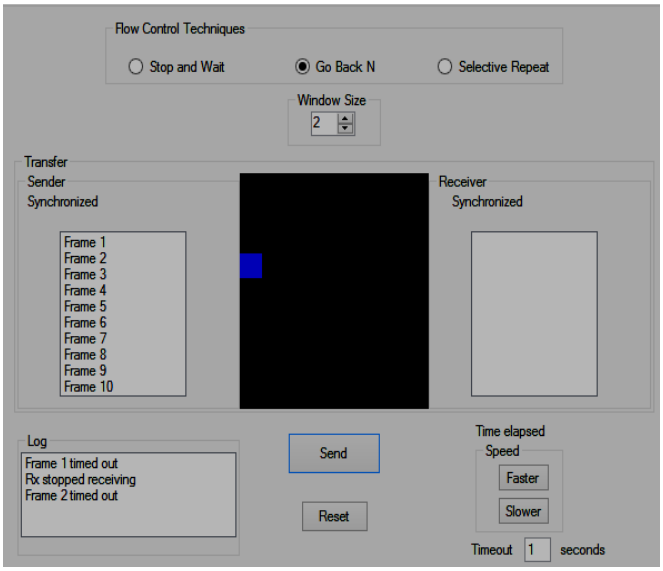

Figure 12.GBN with Timeout of 1 Second

The receiver buffer confirms the same as no frame is currently stored in this buffer. The timeout value was increased to a value of 2 seconds and the observations were the same.

When the timeout was set to 3 seconds, the outcome shown in Figure 13 was obtained. As shown in the log section, frame 1 was successfully detected at the receiver and its ACK was also correctly received at the sender. This is further confirmed by the receiver where it is observed that frame 1 has now been injected into its buffer. In addition, at the sender sliding window, frame 1 has been removed. Unfortunately, the retransmission timer expired before frame 2 could be delivered at the receiver and therefore it is logged as such.

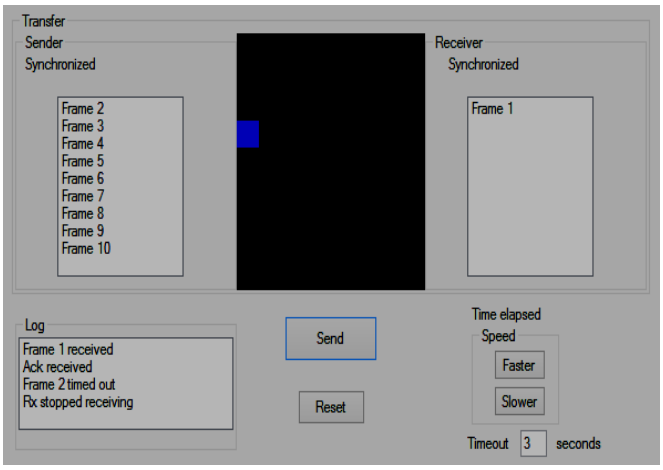

Figure 13.GBN with Timeout of 3 Seconds

When the retransmission timeout was increased to 4 seconds, the outcome obtained is shown in Figure 14. As shown here, both frame 1 and frame 2 have now been successfully received and acknowledged as confirmed by the log section.

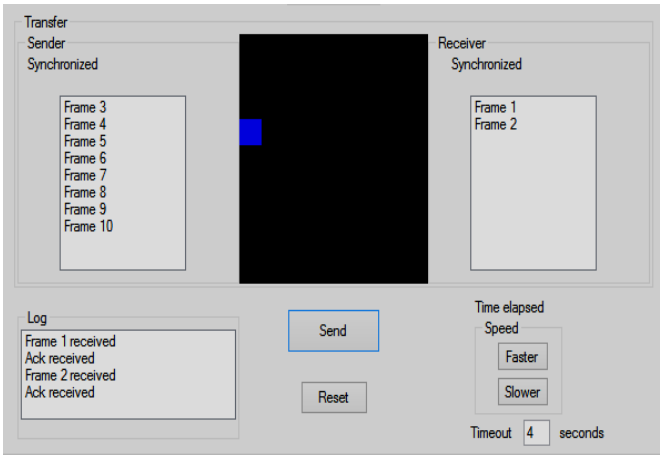

Figure 14.GBN with Timeout of 4 Seconds

It was also noted that at the sender sliding window, these two frames have been removed from the queue, making frame 3 the next candidate frame to be sent. In addition, the receiver buffer has both frame 1 and frame 2 inserted into it.

To investigate GBN frame discarding behaviour when one the frames is received in error, the window size was increased to 10, the retransmission timeout was maintained at 4 seconds and the bandwidth was varied between slower and faster. The results obtained are shown in Figure 15. It was noted that at faster bandwidths, frame 1, frame 2 and

frame 3 were successfully received and acknowledged as confirmed by the receiver buffer, sender sliding window and log section. After frame 3 was received and ACKed, the bandwidth was shifted to slower, making the retransmission timer to expire before frame 4 could be delivered. At this point, the receiver (Rx) stopped receiving further frames and when the bandwidth was switched to faster, no additional frames were correctly detected at the receiver buffer.

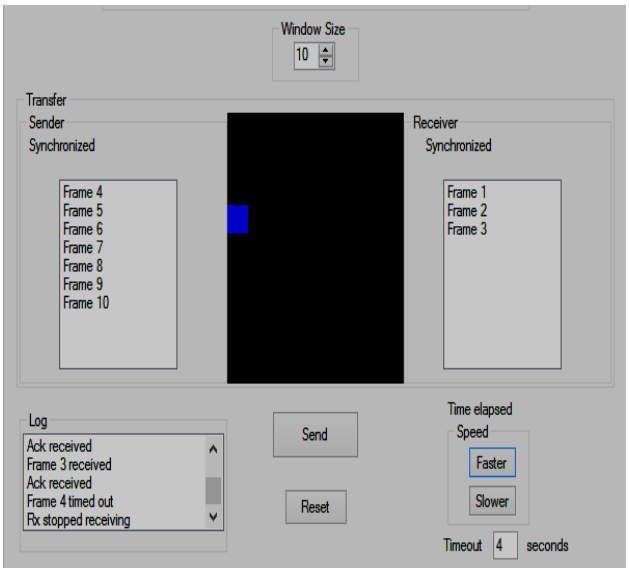

Figure 15: GBN with Timeout of 4 Seconds and Varying Network Bandwidths

This confirms the principle that when GBN protocol encounters frame loss, to ensure in-order delivery, the earliest lost SN must be retransmitted together with the ones following it before new frames could be admitted at the receiver buffer. The next ARQ protocol that was investigated was SR-ARQ. To start with, the window size was set to 2 frames and the retransmission timeout was set to 1 second and the results obtained are shown in Figure 16.

This figure demonstrates that unlike the outcome of Figure 12 where after timeout the receiver refuses to accept additional frames until the lost frame and the ones following it are retransmitted, in SR-ARQ the receiver accepts additional frames after timeout and buffers them in a sequencing queue.

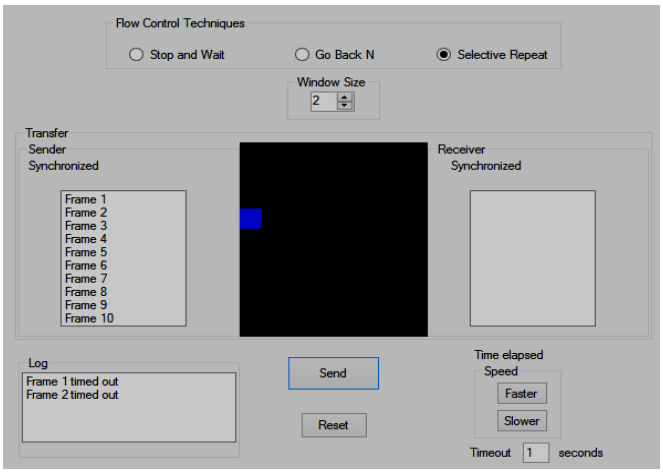

Figure 16.SR-ARQ with Timeout of 1 Second

This is why the log section contents of Figure 12 and Figure 16 differ such that in Figure 12, the RX stopped receiving frames while in Figure 16, the RX continues to admit additional frames despite the fact that a timeout has occurred.

When the retransmission timeout was incremented to 2 seconds, the same results were observed. However, when the retransmission timeout was set to 3 seconds, all the frames were received and acknowledged as shown in Figure 17.

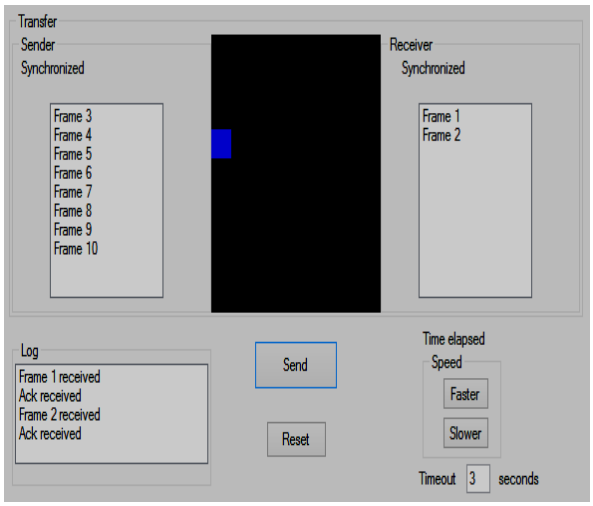

Figure 17.SR-ARQ with Timeout of 3 Seconds

From Figure 17, the receiver buffer indicates that both frame 1 and frame 2 have been received while the sender sliding window signifies that both frame 1 and frame 2 have now been removed from the queue, with frame 3 being the next candidate for transmission. Further, the logs confirm that indeed

both frame 1 and frame 1 have been received and acknowledged. To find out how SR-ARQ behaves when part of the frames are timeout, the sliding window was set to 10 frames while the retransmission timeout was maintained at 3 seconds. The bandwidth was then varied between slower and faster. The outcome obtained is represented in Figure 18.

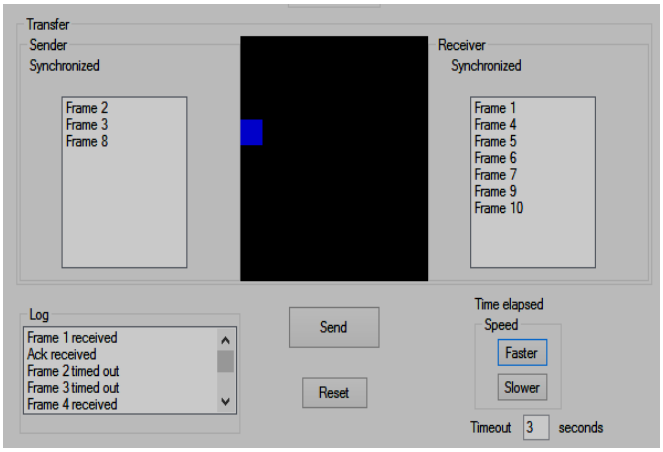

Figure 18.SR-ARQ with Timeout of 3 Seconds and Varying Network Bandwidths

Figure 18 demonstrates farem1, frame 4, frame 5, frame 6, frame 7, frame 9 and frame 10 were successfully received and acknowledged. On the other hand, frame 2, frame 3 and frame 8 were timeout and hence require some retransmissions.

As shown in the sender sliding window, frame2, frame 3 and frame 8 are still in the queue while the rest of the frames have been removed from the queue and instead appear at the receiver buffer. Unlike in GBN where the receiver stopped receiving other frames upon frame loss, SR-ARQ receives and buffers the out-of-order frames and only requires the retransmission of the lost frames.

VIII. PROPOSED ARQ PROTOCOL

A review of the current ARQ protocols has revealed that they fall short of operational efficiency in one way or the other. To address some of these challenges, dual asynchronous prioritized retransmission ARQ (DAPR-ARQ) protocol is proposed. This requires the maintenance of an extra retransmission buffer for storing the frames whose retransmissions have been requested by the destination machine through NACK. This buffer is designed in such a way that one retransmission is performed instantaneously the receiver makes such a request while the other is queued and resent later but before any fresh frames can be sent over the link, hence the name dual asynchronous.

In this protocol, frame retransmissions are prioritized in such a way that no fresh frames are sent until all the NACKed frames are exhausted from the retransmission buffer. In this buffer, frames are released to the communication link on a first come first out (FIFO) basis. This effectively prevents the sender from inserting extra fresh frames into an already lossy communication channel. Although this protocol reduces the overall network throughput, it increases the likelihood of a frame being received correctly.

The premise of the proposed protocol is that many time sensitive video streaming services necessitate an in-order packet delivery in which a frame can be utilized at the application layer only after all the preceding frames in the flow have been received correctly. This protocol reduces the delivery delay since the adaptive increment of the number of retransmissions raise the possibility of correcting the erroneous frames. Moreover, it condenses the number of pending frames as a result of prioritization of retransmissions over new transmissions. Effectively, this prevents the introduction of potentially erroneous frames into the link.

IX. CONCLUSIONS

This paper sough to investigate the effects of retransmission timeout, window size and network bandwidth on the three most common ARQ protocols namely the SAW, GBN-ARQ and SR-ARQ, with special interest in time sensitive smart phone

video streaming services such as Youtube and Netflix. The results obtained showed that the ARQ window size determined the number of frames to be sent to the receiver before it could pause and this applied to both SR-ARQ and GBN. However, for the case of SAW, the window size was always unity as only one frame was permitted at a time during which the sender could pause and wait for an acknowledgement from the receiver. It was also observed that network bandwidth affected the frame transmission process whereby high bandwidth led to higher rates of successful frame delivery and hence high throughput while lower bandwidths led to frequent timeouts and hence lower throughputs. SAW was observed to be the most inefficient due to requirements for pauses after each frame is sent. On the other hand, SR-ARQ was observed to be the most efficient due to continuous transmissions and acknowledgements and the need to resend only damaged or timed out frames. GBN was observed to lead to retransmissions of all frames starting with the one that was reported to be damaged or timed out and hence leads to bandwidth wastage. Based on the weaknesses noted in the current ARQ schemes, a new dual asynchronous prioritized retransmission ARQ protocol has been proposed as a possible solution. Future works in this area involves the statistical and empirical implementation and evaluation of this protocol.

X. REFERENCES

[1]. Swapna B., Eryilmaz A., & Shroff N. (2013). Throughput-Delay Analysis of Random Linear Network Coding for Wireless Broadcasting. IEEE Trans. on Information Theory. Vol. 59, pp. 6328–6341.

[2]. Saleh F. (2016). Automatic Repeat Request (ARQ). Springer Briefs in Electrical and Computer Engineering book series. Pp. 17-35.

[3]. Zhu C., Huo Y., Zhang B., Zhang R., El-Hajjar M., &Hanzo L. (2016). Adaptive-truncated-HARQ-aided layered video streaming relying on interlayer FEC coding. IEEE Transactions on Vehicular Technology. Vol. 65, No. 3, pp. 1506–1521.

[4]. Lai J., Dutkiewicz E., Liu R., &Vesilo R. (2015). Opportunistic spectrum access with two channel sensing in cognitive radio networks. IEEE Trans. Mobile Comput. Vol. 14, No. 1, pp. 126138, Jan. 2015.

[5]. Park E., Song M., Choi W., &Im G. (2016). Maximization of long term average throughput for cooperative secondary system with HARQ based primary system in cognitive radio network. IEEE Commun. Lett. Vol. 20, No. 2, pp. 356-359.

[6]. Ngo H. &Hanzo L. (2014). Hybrid automatic-repeat request systems for cooperative wireless communications. IEEE Communications Surveys Tutorials. Vol. 16, No. 1, pp. 25–45.

[7]. Zongsheng Z., &Qihui W., &Jinlong W. (2015). ARQ Protocols in Cognitive Decode-and-Forward Relay Networks: Opportunities Gain. Radio Engineering. Vol. 24, No. 1, pp. 296- 304.

[8]. Liang W., Nguyen H., &Hanzo L. (2016). Adaptive-TTCM-aided near-instantaneously adaptive dynamic network coding for cooperative cognitive radio networks. IEEE Trans. Veh. Technol. Vol. 65, No. 3, pp. 1314-1325.

[9]. Makki B., Eriksson T., &Svensson T. (2016). On the performance of the relay-ARQ networks. IEEE Transactions on Vehicular Technology. Vol. 65, No. 4, pp. 2078–2096.

[10]. Patel A., Khan M., Merchant S., Desai U., &Hanzo L. (2016). Achievable rates of underlay-based cognitive radio operating under rate limitation. IEEE Transactions on Vehicular Technology. Vol. 65, No. 9, pp. 7149–7159.

[11]. Rehman A., Dong C., Yang L., &Hanzo L. (2016). Performance of cognitive stop-and-wait hybrid automatic repeat reQuest in the face of imperfect sensing. IEEE Access. Vol. 4, pp. 5489-5508.

[12]. Abubakar B. (2017). Automatic Repeat Request (Arq) Protocols.The International Journal of Engineering and Science. Volume 6 ,Issue , 5 , pp. 64-66.

[13]. Elizabeth M., &Mqhele E. (2016).Performance Evaluation of N-Process Stop and Wait in MIMO

Systems with Transmit Antenna Selection.IEEE WiSPNET 2016 conference. Pp. 957- 961.

[14]. Arjun M., &Khushboo C. (2016). Performance analysis of data link layer protocols with a special emphasis on improving the performance of Stop-and-Wait-ARQ. International Conference on Computing for Sustainable Global Development. Pp. 593-597.

[15]. Khalid A., Huda I., Iyad F., Raed T. (2016).A generic buffer occupancy expression for stop-and-wait hybrid automatic repeat request protocol over unstable channels. Telecommunication Systems. Volume 63, Issue 2, pp 205–221.

[16]. Ateeq R., Varghese A., Lie-Liang Y., And L. (2017). Performance of Cognitive Selective-Repeat Hybrid Automatic Repeat Request.IEEE. Volume 4, pp. 9828- 9846.

[17]. Rehman A., Yang L., &Hanzo L. (2016). Performance of cognitive hybrid automatic repeat reQuest: Go-Back-N. In Proc. IEEE 83rd Veh. Technol. Conf., pp. 1-5.

[18]. Saeed R., & Harish V. (2017). Analysis of Feedback Error in Automatic Repeat request. Nokia - Bell Labs. Pp. 1-23.

[19]. Shamna S. (2015).Data Transmission Using Go-Back-N Protocol. International Journal of Innovative Research in Computer Science and Engineering. Volume 1, Issue 2, pp. 1-3.

[20]. Li S., Zhou Y., Yang X., & Dou Z. (2015). Performance evaluation of multiple relays cooperative GBN-ARQ with limited retransmission. Journal of Systems Engineering and Electronics. Vol. 26, No. 6, pp. 1210–1215.

[21]. Hana H. & Mohamed M.(2016). Packet Communication within a Go-Back-N ARQ System Using Simulink. 4th International Connce on Control Engineering & Information Technology.pp.1-6

[22]. Mingsheng G., Jian L., Wei L., &Ning X. (2016).Delivery Delay Analysis of Selective Repeat ARQ in Underwater Acoustic Communications. International Journal of Sensor Networks and Data Communications. Vol. 5, Issue 1, pp. 1-6.

[23]. Chiti F., Fantacci R., &Tassi A. (2014). Evaluation of the resequencing delay for selective repeat ARQ in TDD-based wireless communication systems. IEEE Trans. Veh. Technol. Vol. 63, No. 5, pp. 2450-2455,

[24]. Amal R., & Maria A. (2014). Error Control System for Parallel Multichannel Using Selective Repeat ARQ. International Journal of Computer Science Engineering and Technology. Vol. 4, Issue 12, pp.353-357.

[25]. Swati A., &Atul N.(2014). Performance Analysis of Selective Reject ARQ with effect on Efficiency. International Journal of Computer Networks and Wireless Communications.Vol.4, No1, pp. 46-50.

[26]. Younghoon W. (2016).Transmission and Combining for Hybrid Automatic Repeat Request in Multiple-Input Multiple-Output Systems. Doctorate Thesis, Oregon State University. Pp. 1-106.

[27]. Yasunari M., &Toshiyasu M. (2016).Theoretical Limit of Type-I Hybrid Selective-repeat ARQ with Finite Receiver Buffer. IEICE. Pp. 582- 585.

[28]. Xu K., Ma W., Zhu L., Xu Y., Gao Y., Zhang D., &Xie W.(2016). NTC-HARQ: Network-turbo-coding based HARQ protocol for wireless broadcasting system. IEEE Transactions on Vehicular Technology. Vol. 64, No. 10, pp. 4633–4644.

[29]. Ateeq U., Lie-Liang Y., &Lajos H. (2016). Delay and Throughput Analysis of Cognitive Go-Back-N HARQ in the Face of Imperfect Sensing. IEEE Vehicular Technology Conference. Pp. 1-24.

[30]. Mingsheng G., Jian L., Wei L., &Ning X. (2016).Delivery Delay Analysis of Selective Repeat ARQ in Underwater Acoustic Communications. International Journal of Sensor Networks and Data Communications. Volume 5, Issue 1, pp. 1-6.

[31]. Francesco C., Romano F., & Andrea T. (2014).Evaluation of the Resequencing Delay for Selective Repeat ARQ in TDD-Based Wireless Communication Systems. IEEE Transactions On Vehicular Technology. Vol. 63, No. 5, pp. 2450-2455.

[32]. Hailin X.,&Ju N. (2015). Performance analysis of two-user cooperative ARQ protocol over Rayleigh fading channel. Journal on Wireless Communications and Networking. Vol. 2015, Issue11, pp.1-6.

[33]. Makki B., Svensson T., Eriksson T., AlouiniM(2014). Coordinated hybrid automatic repeat request. IEEE Commun. Lett. Vol.18, Issue 11, pp. 1975–1978.